บรรณานุกรม

ภาษาไทย

ชนะ โคภารักษ์. ศัพท์ไมโครคอมพิวเตอร์. บ.อัมรินทร์ พริ้นติ้งกรุ๊ป จำกัด, 2533.

ชัชวาลย์ ยนต์หงส์. แนะนำภาษาปาสคาล โดย เทอร์โบปาสคาล. สำนักพิมพ์ โอเดียนสโตร์, 2532.

บุญเลิศ เอี่ยมทัศนา. คู่มือ เทอร์โบปาสคาล รุ่น 4.0-5.0. บ.ซีเอ็ดยูเคชั่น จำกัด, 2532.

พงษ์ระพี เตชพาหพงษ์. แอดวานซ์ เอ็มเอสดอส. บ.ซีเอ็ดยูเคชั่น จำกัด, 2533.

วิจิตร ตัณฑสุทธิ์ ,วันชัย ริจิรวนิช และคิริจันทร์ ทองประเสริฐ. การวิจัย การดำเนินงาน. พิมพ์ครั้งที่ 2. บ. ซีเอ็ดยูเคชั่น จำกัด, 2527.

สันติ ชินานุวัติวงศ์. การควบคุมการสูญเสียเหล็กเสริมคอนกรีตในโครงการก่อสร้าง. วิทยานิพนธ์วิศวกรรมศาสตร์มหาบัณฑิต มหาวิทยาลัยเกษตรศาสตร์, 2532.

สุรศักดิ์ สงวนพงษ์. เทคนิคการเขียนโปรแกรมชั้นสูง แอดวานซ์ เทอร์โบปาลคาล. บ.ซีเอ็ดยูเคชั่น จำกัด, 2532.

ภาษาอังกฤษ

Blakey, L.H. Bar Codes : Prescription for Precision, Performance and Productivity. ASCE Journal of Construction Engineering and Management, Vol 116, No. 3 (1990) : 469-479.

Hu, T.C. Integer Programming and Network Flows. Addison-Wesley Publishing Company,Massachusettes, 1969.

Gass, S.I. An Illustrate Guide to Linear Programming. McGraw-Hill Book Company, New York, 1970.

_____ . Linear Programming : Methods and Applications. Fourth Edition, McGraw-Hill Book Company, New York, 1975.

Litton, C.D. 1977. A Frequency Approach to the One-dimensional Cutting Problem for Carpet Rolls. Operation Research Quart 28 (1977) : 927-938.

Loughborough Consultants Limited, University of Technology Loughborough Leicestershire. Lucid : Overlay Handbook 1974.

Mirza, Q.A.B. A Study on the Wastes an Losses in the Building Construction Industry. Master's thesis, Asian Institue of Technology, Bangkok, Thailand, 1984.

Rasdorf, W.J. and Herbert, M.J. Bar Coding in Construction Engineering. ASCE Journal of Construction Engineering and Management, Vol 116, No 2 (1990) : 261-280.

Raza, S. An Application of the linear Programming Technique in One- and Two-Dimensional Stock Cutting Problems. Master's Thesis, Asian Institute of Technology, Bangkok Thailand, 1983.

Stainton, R.S. The Cutting Stock Problem for the Stockholder of Steel Reinforcement Bars. Operation Research Quart 28 [illegible] 139-149.

Stukhart, G. and Cook, E.L. Bar-Code Standardization in Industrial Construction. ASCE Journal of Construction Engineering and Management, Vol 116, No. 3 (1990) : 417-431.

ภาคผนวก

ภาคผนวก ก

คู่มือการใช้โปรแกรม

ก.1. คำนำ

ในการทำงานเหล็กเสริมคอนกรีต ขั้นตอนที่สำคัญอันหนึ่งคือ การคำนวณการตัดเหล็ก เพื่อทราบว่าเหล็กเส้นที่ตัดนี้ควรตัดให้ได้ความยาวเท่าใดและนำเศษที่เหลือไปตัดอีกครั้งเพื่อให้ได้เหล็กเส้นที่ต้องการความยาวเท่าใด หรือเศษที่เหลือต้องกลายเป็นเศษที่ต้องทิ้งผลลัพธ์ที่ตามมาคือการใช้เหล็กเส้นอย่างมีประสิทธิภาพเหลือเศษเหล็กในการทำงานน้อย ทำให้ต้นทุนงานเหล็กเสริมคอนกรีตลดลง

จากความสำคัญดังกล่าว โปรแกรมจึงถูกจัดทำขึ้นเพื่อนำขั้นตอนการคำนวณนี้มาประยุกต์ในรูปโปรแกรมคอมพิวเตอร์ ที่เป็นโปรแกรมสำเร็จรูป ที่สะดวกในการใช้งานทั้งในด้านการรับข้อมูล และการแสดงผล อีกทั้งให้ผลลัพธ์การคำนวณที่มีความถูกต้องสูงจึงเหมาะสำหรับผู้ประกอบการก่อสร้างทั่วไป หรือผู้ที่เกี่ยวข้อง เพราะสามารถทำความเข้าใจ และนำไปใช้ได้โดยง่าย

ก.2. การเตรียมแผ่นใช้งาน

โปรแกรม YTP มีขนาดของโปรแกรมไม่ใหญ่นัก สามารถเก็บไฟล์ที่ต้องใช้งานได้ภายในแผ่นดิสค์ขนาด 360 KB เพียงแผ่นเดียว และยังมีเนื้อที่เหลือสำหรับเก็บไฟล์ข้อมูลได้อีก สำหรับขั้นตอนการเตรียมแผ่นใช้งานให้เริ่มดังนี้

1) ฟอร์แมตดิสค์ โดยใช้คำสั่ง

A> FORMAT B:/S

เป็นการฟอร์แมตให้มีซิสเต็มด้วย นั่นคือสามารถ BOOT แผ่นได้ด้วยแผ่นของมันเอง หรือ

A> FORMAT B:

เป็นการฟอร์แมตแบบไม่มีซิสเต็ม ซึ่งต้อง BOOT ด้วย DOS ก่อนทำงานทำให้มีเนื้อที่ว่างเหลือบนแผ่นดิสค์มากกว่าแบบแรก

2) ก๊อปปี้ไฟล์ YTP.EXE กับ HA.YT จากมาสเตอร์ดิสค์สู่แผ่นใช้งาน โดยใช้แผ่นมาสเตอร์อยู่ในไดร์ฟ A และแผ่นใช้งานในไดร์ฟ B แล้วใช้คำสั่ง

A> COPY YTP.EXE B:
A> COPY HA.YT B:

จะได้แผ่นดิสค์ที่มีโปรแกรมพร้อมใช้งานในไดร์ฟ B

ก.3. ขั้นตอนการทำงาน

ขั้นตอนการทำงานเริ่มด้วยการถอดแบบเหล็กจากการทำ BARLISTS แล้วนำความยาว ขนาด และชนิดของเหล็กมาป้อนเข้าสู่โปรแกรม โดยโปรแกรม YTP มีรูปแบบข้อมูลดังนี้

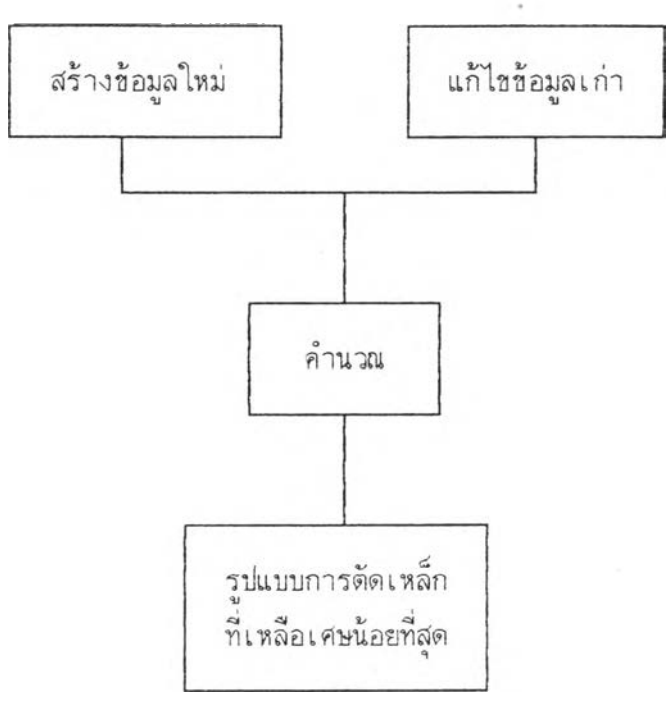
	เบอร์คาน	เบอร์เหล็ก	ความยาว	จำนวน	ขนาด	ชนิด
1. 4 Y32-1-9.65	B1	001	9.65	4	32	DB30
2. 2 Y25-2-6.70	B1	002	6.70	2	25	DB30
3. 4 Y20-1-9.65	B1	003	9.65	4	20	DB30
4. 4 Y32-6-5.00	B1	004	5.00	4	32	DB30
5. 4 Y20-7-3.00	B1	005	3.00	4	20	DB30
6. 4 Y25-10-1.90	B1	006	1.90	4	25	DB30
7. 63 R12-16-2.45	B1	007	2.45	63	12	RB25
8. 115 R12-17-1.98	B1	008	1.98	115	12	RB25

แสดงให้เห็นว่าสามารถเขียนเป็นรูปแบบข้อมูลเพื่อป้อนเข้าสู่โปรแกรมได้หลายแบบจากตัวอย่างนี้ยกมาเพียงสองแบบ รูปแบบแรกเป็นการเขียนโดยแปลงหมายเลขในเบอร์เหล็กให้เรียงเป็นแบบ RUNNING NUMBER เพื่อสะดวกในการดู ต่อเติม และแก้ไขข้อมูล และให้เบอร์คานเป็นเบอร์ B1 ซึ่งอาจไม่สะดวกในการที่ต้องเอาผลจากการคำนวณมาเทียบหาเบอร์เหล็กตามแบบอีกครั้ง

การป้อนข้อมูล โปรแกรม YTP มีรูปแบบการป้อนค่าข้อมูลแบบ FULL SCREEN คือสามารถให้ค่าข้อมูลได้ทั้งหน้าจอภาพ เพียงแต่เลื่อนเคอร์เซอร์ไปยังตำแหน่งที่ต้องการแก้ไขแล้วพิมพ์ค่าใหม่ทับเข้าไป เมื่อป้อนข้อมูลเรียบร้อย ให้กด ENTER เป็นการสิ้นสุดการป้อนข้อมูล

ขั้นตอนต่อไปคือการคำนวณ ให้เลือกฟังก์ชั่น RUN เพื่อทำการคำนวณ โดยใช้การจัดกลุ่มหารูปแบบเหล็กที่เป็นไปได้ แล้วมาหารูปแบบที่ประหยัดเหล็กที่สุดด้วยระบบสมการเชิงเส้นตรง ได้ผลลัพธ์ออกมาในรูปของรูปแบบการตัดเหล็กเส้นที่เหลือเศษน้อยที่สุด

สามารถแสดงเป็นแผนภาพได้ดังนี้



ในขั้นตอนการคำนวณนี้ โปรแกรม YTP ได้ถูกแบบให้สามารถคำนวณได้จากเหล็กเส้นความยาวมาตรฐานจำนวน 2 ความยาว ที่แก้ไขได้ และจากเศษเหล็กมาตรฐานจำนวน 2 ความยาวเช่นกัน

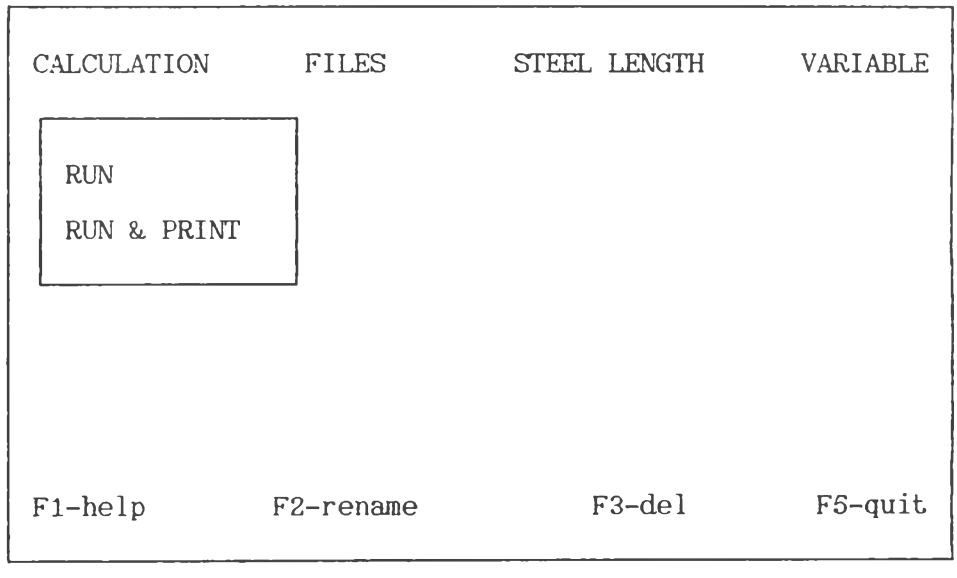
ส่วนผลลัพธ์นั้นนอกจากได้รูปแบบการตัดเหล็กที่เหลือเศษน้อยที่สุดแล้วยังมีเบอร์เหล็กติดมาด้วยทำให้ทราบว่าเหล็กเส้นที่ได้นี้ควรนำไปใช้กับองค์อาคารส่วนใด

ก.4. วิธีการใช้โปรแกรม

ก.4.1. ฟังก์ชั่นใช้งานหลัก

ประกอบด้วย

ก.4.1.1. CALCULATION เป็นคำสั่งให้คำนวณ ดังรูปที่ ก.1 แสดงจอภาพเมื่อใช้ฟังก์ชั่น CALCULATION

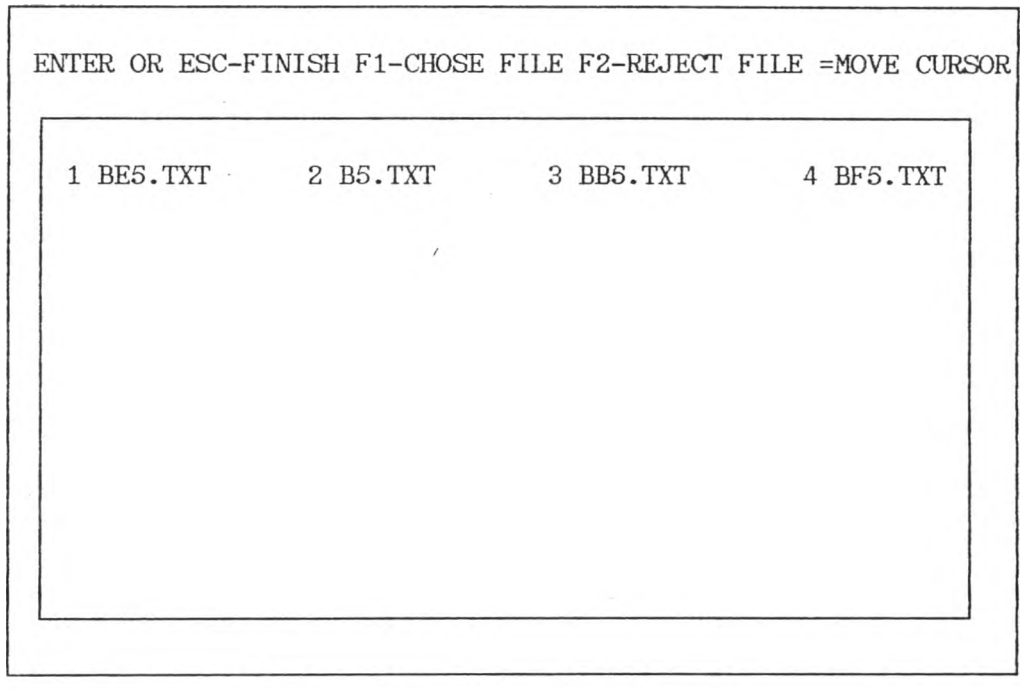


รูปที่ ก.1

จากจอภาพปรากฏฟังก์ชั่นย่อยให้เลือก 2 ฟังก์ชั่น ดังนี้

- RUN คือการคำนวณโดยแสดงผลทางจอภาพ
- RUN & PRINT คือการคำนวณโดยการแสดงผลทางเครื่องพิมพ์

การเลือกใช้ทำได้โดยเลื่อนลูกศรขึ้นลงเมื่ออยู่ในตำแหน่งที่ต้องการให้กด ENTER จะปรากฏจอภาพดังรูป ก.2



รูปที่ ก.2

จากจอภาพปรากฏคำอธิบายฟังก์ชั่น และช่องหน้าต่างย่อย ภายในช่องหน้าต่างแสดงชื่อไฟล์ข้อมูล (*.TXT) และมีแถบ CURSOR สว่างชี้ไปที่ชื่อไฟล์ตัวแรก เป็นการแสดงไฟล์ที่ถูกเลือกใช้ในการคำนวณ การเลื่อนแถบ CURSOR ทำได้โดยใช้ลูกศรในการเลื่อนไปมาเมื่อต้องการใช้ไฟล์ใดในการคำนวณให้กด F1 เมื่อเลื่อน CURSOR ไปตำแหน่งอื่นแถบสว่างจะยังปรากฏเป็นการแสดงไฟล์ที่ถูกเลือกนั้น การเลือกไฟล์คำนวณสามารถจะเลือกไฟล์ในการคำนวณได้ครั้งละหลาย ๆ ไฟล์ ส่วนการยกเลิกไฟล์สามารถทำได้โดยเลื่อน CURSOR ไปที่ไฟล์ที่ต้องการยกเลิกการคำนวณแล้วกด F2 แถบ CURSOR ที่เคยเรืองแสงจะดับไป เมื่อทำการเลือกไฟล์ที่ต้องการแล้วให้กด "ENTER" หรือ "ESC" เพื่อเข้าสู่ขั้นตอนการคำนวณ ฟังก์ชั่นดังกล่าวข้างต้นแสดงอยู่ในคำอธิบายความหมายอยู่บริเวณด้านบนของจอภาพ ในขั้นตอนการคำนวณเริ่มด้วยการ แสดงข้อมูลในไฟล์ที่เลือกนั้นจนหมดแล้วหยุดรอให้ป้อนค่าความยาวเหล็กมาตรฐานหรือความยาวเหล็กที่เหลืออยู่สำหรับตัดซึ่งถ้าไม่แก้ไขโปรแกรมจะใช้ข้อมูลจากค่าที่กำหนดไว้ในดอนแรกตามที่ปรากฏอยู่ด้านล่างของจอภาพ ดังรูป ก.3

ENTER OR ESC-FINISH F1-CHOSE FILE F2-REJECT FILE =MOVE CURSOR

1	B1	001	2219	10	9	RB24
2	B4	014	2219	7	9	RB24
3	B4	013	2219	9	9	RB24
4	B3	010	2219	6	9	RB24
5	B3	009	2219	6	9	RB24
6	B2	006	2219	10	9	RB24
7	B1	002	2219	8	9	RB24
8	B3	008	3083	2	16	RB24
9	B4	012	3360	1	15	RB24
10	B2	004	3959	1	15	DB30
11	B2	007	2219	8	12	DB30
12	B2	005	2913	1	12	RB24
13	B4	011	3288	2	12	RB24
14	B2	003	3887	2	12	RB24
15	BEAM	015	2955	4	25	DB30
16	BEAM	015	10000	4	25	DB30
17	BEAM	015	10000	4	25	DB30

F1-1SL.12.0 F2-2SL.10.0 F3-RL.0.0Q.0 F4-2RL.0.0Q.0 ENTER-READY

รูปที่ ก.3

F1 บอกค่าความยาวเหล็กมาตรฐานเส้นที่ 1

F2 บอกค่าความยาวเหล็กมาตรฐานเส้นที่ 2

F3 บอกค่าความยาวเศษเหล็กมาตรฐานเส้นที่ 1 และจำนวนที่เหลือ

F4 บอกค่าความยาวเศษเหล็กมาตรฐานเส้นที่ 2 และจำนวนที่เหลือ

ค่าในฟังก์ชั่น F1 - F4 นี้เป็นฟังก์ชั่นสำหรับกำหนดค่าความยาวเหล็กมาตรฐานที่มีกำหนดไว้เดิมแล้วในโปรแกรม แต่สามารถเปลี่ยนแปลงได้โดยใช้ฟังก์ชั่นเหล่านี้ การเปลี่ยนแปลงค่าในขั้นตอนนี้เป็นลักษณะการให้ค่าแบบชั่วคราว คือใช้เป็นค่าในการคำนวณสำหรับข้อมูลชุดที่ปรากฏบนจอภาพขณะนั้นเท่านั้น เมื่อทำการเปลี่ยนค่าความยาวเหล็กเรียบร้อยแล้วให้กด "ENTER" เพื่อเข้าสู่การคำนวณขั้นต่อไป

DIAMETER 12 RB24

qual.*length	qual.*length	qual.*length	qual.*length	no.cut	st
3*2.21 B2 007 3	1*3.28 B4 011 1			1	10
1*2.21 B2 007 1	2*3.88 B2 003 2			1	10
1*2.91 B2 005 1	4*2.21 B2 007 4			1	12
1*3.28 B4 011 1					
REST STEEL = 7.09 USE 10.00 M. = 3 USE 12.00 M. = 1					

F1-1SL.12.0 F2-2SL.10.0 F3-RL.0.0Q.0 F4-2RL.0.0Q.0 ENTER-READY

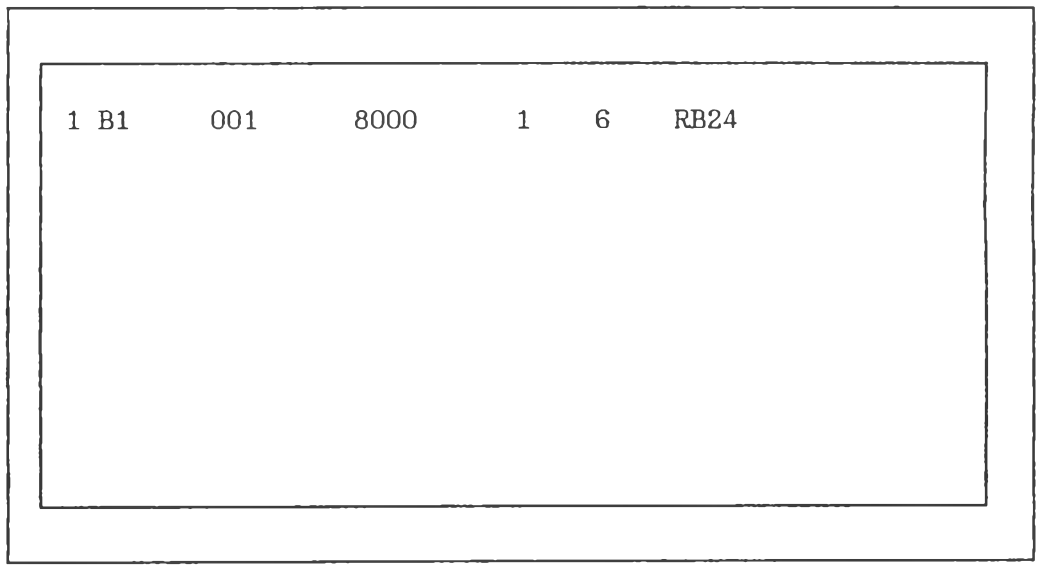
รูปที่ ก.4

รูปที่ ก.4 แสดงจอภาพของผลการคำนวณ จากตัวอย่างมีความหมายดังนี้

บริเวณตรงกลางด้านบนกรอบบอกถึงขนาดเส้นผ่าศูนย์กลางเหล็กเส้น และชนิดของเหล็กนั้นเช่นตามจอภาพดังรูป ก.5 หมายความว่ากำลังคำนวณเหล็กเส้นผ่าศูนย์กลาง 12 มม. ชนิด RB 24

ในทำนองเดียวกันโปรแกรมจะรอการแก้ไขค่าความยาวเหล็กมาตรฐาน ถ้าเปลี่ยนเรียบร้อยแล้วหรือไม่เปลี่ยนแปลงให้กด "ENTER" การทำงานจะเป็นลักษณะนี้จนสิ้นสุดแล้วกลับมาสู่หน้าจอเดิมตามรูป ก.1

ก.4.1.2. FILE เป็นฟังก์ชั่นเกี่ยวกับการจัดการข้อมูล มีฟังก์ชั่นย่อยเพียงฟังก์ชั่นเดียวคือ "DATA FILE" เมื่อกด "ENTER" โปรแกรมจะถามชื่อไฟล์ ด้วยคำถาม "ENTER FILES NAME" ชื่อไฟล์นี้มีความยาวไม่เกิน 7 ตัวอักษร และไม่มีนามสกุลกล่าวคือไม่ต้องมี "-.*" โปรแกรมจะกำหนดนามสกุล .TXT ให้โดยอัตโนมัติ



รูปที่ ก.5

รูปที่ ก.5 แสดงจอภาพภายหลังให้ชื่อไฟล์ และกด "ENTER" เรียบร้อยแล้ว ในกรณีที่ป้อนชื่อไฟล์ใหม่จะมีข้อมูลที่ปรากฏเป็นข้อมูลตัวอย่างให้ 1 บรรทัด ประกอบด้วย

ชุดแรก คือ เบอร์เหล็ก
ชุดที่สอง คือ RUNNING NUMBER ให้ค่าเพิ่มเรื่อย ๆ เมื่อขึ้นข้อมูลบรรทัดใหม่
ชุดที่สาม คือ ความยาวของเหล็กมีหน่วยเป็นเซ็นติเมตร

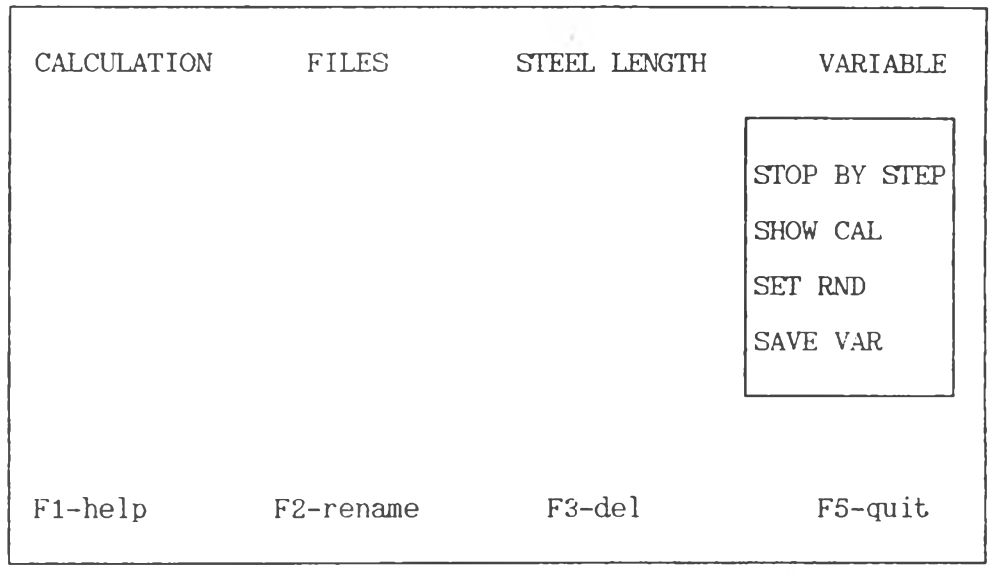
ชุดที่สี่ คือ จำนวนของข้อมูลกลุ่มนี้
ชุดที่ห้า คือ เส้นผ่าศูนย์กลาง
ชุดสุดท้ายคือ ชนิดของเหล็ก

สำหรับข้อมูลกลุ่มต่อ ๆ ไป โปรแกรมจะคัดลอกข้อมูลจากข้อมูลกลุ่มที่ผ่านมาเพื่อความรวดเร็วในการป้อนข้อมูล เมื่อป้อนหรือแก้ข้อมูลเสร็จ แล้วกด "ENTER" มีคำถามว่าจะเก็บข้อมูลหรือไม่ "SAVE YOUR DATA" ถ้ากด Y แสดงว่าเก็บข้อมูลไว้ ถ้ากดคีย์ใด ๆ แสดงว่าไม่เก็บไฟล์ที่เขียนใหม่นี้ จากนั้นกลับคืนสู่เมนเมนูเดิม ในโปรแกรมมีคีย์พิเศษช่วยในการป้อน ลบ และแก้ไขข้อมูลดังนี้

- F5 เป็นคีย์คำสั่งยกเลิกบล็อค
- F6 เป็นคีย์คำสั่งลบข้อความในบล็อค
- F7 เป็นคีย์คำสั่งหมายต้นบล็อค
- F8 เป็นคีย์คำสั่งหมายท้ายบล็อค
- F9 เป็นคีย์คำสั่งคัดลอกบล็อค

ก.4.1.3. STANDARD LENGTH เป็นฟังก์ชั่นการให้ค่าความยาวเหล็กเส้นมาตรฐานโดยมีให้เลือกกำหนดได้สองค่าความยาว การเก็บค่าในขั้นตอนนี้เป็นการเก็บในลักษณะการติดตั้งค่ากึ่งถาวรโปรแกรมจะเก็บค่านี้ไว้ตลอดการทำงานของโปรแกรม ในการติดตั้งแบบถาวรสามารถทำได้โดยใช้ฟังก์ชั่น "SAVE VAR" ในเมนฟังก์ชั่น "OTHER" ซึ่งกล่าวไว้แล้วในหัวข้อ ก.1.4. เมื่อเลื่อนแถบ CURSOR มาที่ฟังก์ชั่นนี้แล้วกด "ENTER" จะปรากฏคำถามให้ป้อนค่าความยาวเหล็กมาตรฐานที่ต้องการสองค่า เมื่อป้อนค่าความยาวเหล็กมาตรฐานเรียบร้อยแล้วให้กด " ENTER" ในกรณีต้องการใช้ความยาวเหล็กมาตรฐานเพียงความยาวเดียวให้กำหนดความเหล็กอีกเส้นเป็น "0" เมื่อเรากด "ESC" ก็จะกลับสู่เมนเมนู

ก.4.1.4. OTHER เป็นฟังก์ชั่นที่เก็บค่าฟังก์ชั่นพิเศษต่าง ๆ ซึ่งโดยทั่วไปไม่ควรเปลี่ยนแปลงแก้ไขค่าใด ๆ ในฟังก์ชั่นกลุ่มนี้ เมื่อเลื่อนแถบ CURSOR มาที่ฟังก์ชั่นนี้แล้วกด "ENTER" จะปรากฏจอภาพ ดังรูป ก.6



รูปที่ ก.6

ปรากฏฟังก์ชั่นย่อยในหน้าต่างดังนี้

- "STOP BY STEP" คือการหยุดจอภาพขณะทำงานเพื่อดูค่าคำตอบการจัดเหล็กทีละกลุ่ม ซึ่งจะรอการกด "ENTER"ทุกขั้นตอน
- "SHOW CAL" คือการแสดงขั้นตอนการคำนวณตามระบบสมการเชิงเส้นตรงเพื่อผู้ใช้พิจารณาเซตค่า"RND" ซึ่งกล่าวถึงในหัวข้อถัดไป โดยอาจทำให้ได้ค่าการคำนวณที่ดีขึ้น
- "SET RND" คือการตั้งค่า "RND" ซึ่งเป็นค่าต่ำสุดที่ปัดค่าคำตอบในการคำนวณจากทศนิยมเป็นเลขจำนวนเต็มในขั้นตอนการคำนวณสมการเชิงเส้นตรง ในโปรแกรมได้กำหนดค่าที่เหมาะสมที่สุดไว้แล้ว ผู้ใช้สามารถอ่านรายละเอียดเพื่อความเข้าใจที่ดีขึ้นได้จากวิทยานิพนธ์
- "SAVE VAR" คือการเก็บค่าแบบถาวรของค่าตัวแปรสามฟังก์ชั่นข้างบนและฟังก์ชั่น STANDARD LENGTH ในทำนองเดียวกันเมื่อกด ESC จะกลับสู่เมนเมนู

ก.4.2. ฟังก์ชั่นช่วย

-F1	HELP	เป็นคำสั่งเพื่อแสดงข้อความอธิบายการทำงานของโปรแกรม
-F2	RENAME	เป็นคำสั่งเพื่อเปลี่ยนชื่อไฟล์ข้อมูล
-F3	DEL	เป็นคำสั่งเพื่อลบไฟล์ข้อมูลที่ไม่ต้องการใช้แล้วออก
-F10	QUIT	เป็นคำสั่งเพื่อออกจากโปรแกรม YTP

ก.5. การจัดสรรหน่วยความจำ

โปรแกรม YTP เหมือนโปรแกรมสำเร็จรูปทั่วไป ที่ต้องมีความจำกัดในด้านการจัดสรรหน่วยความจำในตัวแปรต่างๆในโปรแกรม ซึ่งโปรแกรม YTP ได้จัดสรรหน่วยความจำให้ได้สูงสุด และตรงตามความจำเป็นที่ต้องใช้ตัวแปรเหล่านั้น ดังจะกล่าวเฉพาะที่สำคัญดังนี้

- ความยาวเหล็ก และเศษเหล็กมาตรฐานมีได้ไม่เกินสองความยาว
- ความยาวเหล็ก และเศษเหล็กมาตรฐานมีความยาวได้ไม่เกิน 99 เมตร
- ข้อมูลแต่ละชุดข้อมูลมีได้ไม่เกิน 4 ตัวอักษร
- ความยาวเหล็กเส้นในแต่ละกลุ่มข้อมูลมีความยาวได้ไม่เกิน 99 เมตร
- จำนวนเหล็กเส้นในแต่ละกลุ่มข้อมูลมีได้ไม่เกิน 9999 เส้น
- จำนวนกลุ่มข้อมูลมีได้ไม่เกิน 600 กลุ่มข้อมูล ต่อหนึ่งไฟล์ข้อมูล ต่อการคำนวณหนึ่งครั้ง
- ชื่อไฟล์ข้อมูลไม่ต้องมีนามสกุล และมีความยาวไม่เกิน 7 ตัวอักษร

ภาคผนวก ข

แสดงผลลัพธ์เพิ่มเติม

ตัวอย่างที่เพิ่มเติมในหัวข้อนี้ เพื่อหาค่าเฉลี่ยของผลลัพธ์ในการคำนวณ โดยมีตัวอย่างจากข้อมูลทดสอบของ นายสันติ ชินานุวัติวงศ์ ข้อมูลในส่วนของพื้นและเสาจากโครงการอาคารที่จอดรถ ตึกเมืองไทยประกันชีวิต ของ บ. ส.เสนีย์ และข้อมูลจากโครงการ LOW RISE CONDOMINIUM ของ บ.เอส เอส ชวน

ตัวอย่างที่ ช.1

ข้อมูล

S1	001	2820	72	25	DB30
S1	002	2980	2	25	DB30
S1	003	3020	72	25	DB30
S1	004	3500	228	25	DB30
S1	005	3560	9	25	DB30
S1	006	3880	24	25	DB30
S1	007	3910	12	25	DB30
S1	008	4000	72	25	DB30
S1	009	6060	6	25	DB30
S1	010	6240	12	25	DB30
S1	011	6320	15	25	DB30
S1	012	6560	15	25	DB30
S1	013	8410	27	25	DB30
S1	014	9110	9	25	DB30
S1	015	9260	9	25	DB30
S1	016	11700	6	25	DB30
S1	017	11760	6	25	DB30
S1	018	11870	9	25	DB30

ผลลัพธ์ที่ได้

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*2.82	1*9.11			9	12.0
S1 001 9	S1 014 9				
1*2.98	2*3.50			2	10.0
S1 002 2	S1 004 4				
1*3.02	1*3.50			28	10.0
S1 003 56	S1 004 4				
3*3.50				8	12.0
S1 004 24					
1*3.56	1*6.24			28	10.0
S1 005 9	S1 010 9				
3*3.88				8	12.0
S1 006 24					
3*3.91				2	12.0
S1 007 6					
3*4.00				24	12.0
S1 008 72					
1*3.91	1*6.06			6	10.0
S1 007 6	S1 009 6				
1*3.50	1*6.24			3	10.0
S1 004 3	S1 010 3				
1*2.82	2*3.50			63	10.0
S1 001 63	S1 004 126				
1*3.02	1*6.56			15	10.0
S1 003 15	S1 012 15				
1*3.50	1*8.41			27	12.0
S1 004 27	S1 013 27				
1*3.50	1*6.32			15	10.0
S1 004 15	S1 011 15				

DIAMETER 25 DB30					
quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*9.26				9	10.0
S1 015 9					
1*11.70				6	12.0
S1 016 6					
1*11.76				6	12.0
S1 017 6					
1*11.87				9	12.0
S1 018 9					
1*3.02	1*3.50			1	10.0
S1 003 1	S1 004 1				
REST STEEL = 69.05					
USE 10.00 M. = 151					
USE 12.00 M. = 99					

ตัวอย่างที่ ข.2

ข้อมูล

B1	001	1240	720	20	DB30
B1	002	2990	602	20	DB30
B1	003	3000	64	20	DB30
B1	004	3250	560	20	DB30
B1	005	3300	220	20	DB30
B1	006	3350	268	20	DB30
B1	007	3360	320	20	DB30
B1	008	3430	156	20	DB30
B1	009	4050	220	20	DB30
B1	010	4300	776	20	DB30

ผลลัพธ์ที่ได้

DIAMETER 20 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
4*1.24	1*2.99	1*4.05		32	12.0
B1 001 128	B1 002 32	B1 009 32			
4*2.99				108	12.0
B1 002 432					
2*1.24	1*3.00	2*3.25		138	12.0
B1 001 128	B1 003 64	B1 004 128			
2*1.24	1*2.99	2*3.25		138	12.0
B1 001 276	B1 002 138	B1 004 276			
3*3.30				80	12.0
B1 005 150					
1*1.24	2*3.35	1*4.05		100	12.0
B1 001 100	B1 006 200	B1 009 100			
1*3.36	2*4.30			320	12.0
B1 007 320	B1 010 640				
1*3.25	1*3.30	1*3.43		68	10.0
B1 004 68	B1 005 68	B1 008 68			
1*1.24	1*3.25	1*3.43	1*4.05	88	12.0
B1 001 88	B1 004 88	B1 008 88	B1 009 88		
1*3.35	2*4.30			68	12.0
B1 006 68	B1 010 136				
2*3.30				1	10.0
B1 005 2					
REST STEEL = 39.24					
USE 10.00 M. = 119					
USE 12.00 M. = 918					

ตัวอย่างที่ ข.3

ข้อมูล

S1	001	10000	1383	16	DB30
S1	002	9350	13	16	DB30
S1	003	8300	15	16	DB30
S1	004	7400	13	16	DB30
S1	005	7250	14	16	DB30
S1	006	7200	6	16	DB30
S1	007	7300	103	16	DB30
S1	008	5300	14	16	DB30
S1	009	5550	103	16	DB30
S1	010	6000	10	16	DB30
S1	011	5000	6	16	DB30
S1	012	2500	7	16	DB30
S1	013	4000	10	16	DB30
S1	014	3300	30	16	DB30
S1	015	800	453	16	DB30

ผลลัพธ์ที่ได้

DIAMETER 16 DB30					
quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*0.80	2*5.55			51	12.0
S1 015 51	S1 009 102				
1*2.50	1*7.25			7	10.0
S1 012 7	S1 005 7				
3*4.00				1	12.0
S1 013 3					
2*5.00				3	10.0
S1 011 6					

DIAMETER 16 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*3.30	1*5.30			14	12.0
S1 014 28	S1 008 14				
1*3.30	1*8.30			1	12.0
S1 014 1	S1 003 1				
2*6.00				5	12.0
S1 010 10					
1*0.80	1*4.00	1*7.20		6	12.0
S1 015 6	S1 013 6	S1 006 6			
3*0.80	1*7.25			7	10.0
S1 015 21	S1 005 7				
3*0.80	1*7.30			103	10.0
S1 015 309	S1 007 103				
3*0.80	1*7.40			13	10.0
S1 015 39	S1 004 13				
2*0.80	1*8.30			13	10.0
S1 015 26	S1 003 13				
1*9.35				13	10.0
S1 002 13					
1*10.00				1383	10.0
S1 001 1383					
1*3.30	1*8.30			1	12.0
S1 014 1	S1 003 1				
1*0.80	1*4.00	1*5.55		1	12.0
S1 015 1	S1 013 1	S1 009 1			
REST STEEL = 56.40					
USE 10.00 M. = 1542					
USE 12.00 M. = 80					

ตัวอย่างที่ ข.4

ข้อมูล

S1	001	10000	3641	16	DB30
S1	002	9000	7	16	DB30
S1	003	7500	7	16	DB30
S1	004	6500	22	16	DB30
S1	005	5400	89	16	DB30
S1	006	5000	812	16	DB30
S1	007	4500	506	16	DB30
S1	008	4000	690	16	DB30
S1	009	3500	9	16	DB30
S1	010	6000	16	16	DB30
S1	011	3300	16	16	DB30
S1	012	2400	72	16	DB30
S1	013	2000	456	16	DB30
S1	014	1500	7	16	DB30
S1	015	1800	120	16	DB30
S1	016	2300	352	16	DB30
S1	017	6550	10	16	DB30

ผลลัพธ์ที่ได้

DIAMETER 16 DB30					
quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*1.50	1*2.00	1*6.50		7	10.0
S1 014 7	S1 013 7	S1 004 7			
3*1.80	2*2.30			34	12.0
S1 015 102	S1 016 68				
6*2.00				60	12.0
S1 013 360					

DIAMETER 16 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*2.30	1*2.40	1*5.00		52	12.0
S1 016 104	S1 012 52	S1 006 52			
1*1.80	1*2.40	1*3.30	1*4.50	16	12.0
S1 015 16	S1 012 16	S1 011 16	S1 007 16		
1*3.50	1*6.50			9	10.0
S1 009 9	S1 004 9				
3*4.00				230	12.0
S1 008 690					
2*4.50				241	10.0
S1 007 482					
2*5.00				379	10.0
S1 006 758					
1*2.00	2*2.30	1*5.40		89	12.0
S1 013 89	S1 016 178	S1 005 89			
2*6.00				8	12.0
S1 010 16					
1*6.50				6	10.0
S1 004 6					
1*6.55				10	10.0
S1 017 10					
1*4.50	1*7.50			7	12.0
S1 007 7	S1 003 7				
1*9.00				7	10.0
S1 002 7					
1*10.00				3641	10.0
S1 001 3641					
1*2.30	4*2.40			1	12.0
S1 016 1	S1 012 9				

DIAMETER 16 DB30					
quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*1.80	2*5.00			1	12.0
S1 015 1	S1 006 2				
1*1.80	2*2.30	1*4.50		1	10.0
S1 015 1	S1 016 1	S1 007 1			
REST STEEL = 305.02					
USE 10.00 M. = 4335					
USE 12.00 M. = 464					

ตัวอย่างที่ ข.5

ข้อมูล

S1	001	2000	55	25	DB30
S1	002	2700	55	25	DB30
S1	003	4000	20	25	DB30
S1	004	4300	20	25	DB30
S1	005	5000	1434	25	DB30
S1	006	5400	605	25	DB30
S1	007	5500	10	25	DB30
S1	008	5700	22	25	DB30
S1	009	6000	80	25	DB30
S1	010	10000	319	25	DB30
S1	011	8400	324	25	DB30
S1	012	2000	324	25	DB30

ผลลัพธ์ที่ได้

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
6*2.00				59	12.0
S1 001 55					
S1 012 299					
1*2.70	1*8.40			55	12.0
S1 002 55	S1 011 55				
3*4.00				6	12.0
S1 003 18					
1*2.00	1*4.30	1*5.70		20	12.0
S1 012 20	S1 004 20	S1 008 20			
2*5.00				717	10.0
S1 005 1434					
2*5.40				302	12.0
S1 006 604					
2*5.50				302	12.0
S1 007 10					
2*5.70				5	12.0
S1 008 2					
2*6.00				1	12.0
S1 009 80					
1*8.40				269	10.0
S1 011 269					
1*10.00				319	10.0
S1 010 319					
1*2.00	1*4.00	1*5.40		1	12.0
S1 012 1	S1 003 1	S1 006 1			

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
4*2.00 S1 012 4	1*4.00 S1 003 1			1	12.0
REST STEEL = 848.50 USE 10.00 M. = 1305 USE 12.00 M. = 490					

ตัวอย่างที่ ข.6

ข้อมูล

S1	001	1200	303	12	DB30
S1	002	6500	21	12	DB30
S1	003	2500	105	12	DB30
S1	004	1200	88	12	DB30
S1	005	2400	25	12	DB30
S1	006	3000	25	12	DB30

ผลลัพธ์ที่ได้

DIAMETER 12 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
10*1.20				39	12.0
S1 001 303					
S1 004 87					
5*2.40				5	12.0
S1 005 25					
4*2.50				21	10.0
S1 003 84					
4*3.00				1	12.0
S1 006 4					
1*2.50	1*3.00	1*6.50		21	12.0
S1 003 21	S1 006 21	S1 002 21			
1*1.20				1	10.0
S1 004 1					
REST STEEL = 8.8					
USE 10.00 M. = 22					
USE 12.00 M. = 66					

ตัวอย่างที่ ข.7

ข้อมูล

S1	001	2500	414	12	DB30
S1	002	3300	41	12	DB30
S1	003	1200	260	12	DB30
S1	004	10000	18	12	DB30

ผลลัพธ์ที่ได้

DIAMETER 12 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
10*1.20				21	12.0
S1 003 210					
5*2.50				72	10.0
S1 001 288					
1*1.20	3*2.50	1*3.30		41	12.0
S1 003 41	S1 001 123	S1 002 41			
1*10.00				18	10.0
S1 004 18					
3*1.20	3*2.50			1	12.0
S1 003 3	S1 001 3				
6*1.20				1	12.0
S1 003 6					
REST STEEL = 3.7					
USE 10.00 M. = 91					
USE 12.00 M. = 63					

ตัวอย่างที่ ข.8,ข9,ข10,ข11

ข้อมูล

S1	001	3000	27	12	DB30
S1	002	2600	18	12	DB30
S1	003	1000	6	12	DB30
S1	004	2400	119	16	DB30
S1	005	5000	16	16	DB30
S1	006	4000	176	16	DB30
S1	007	2000	21	20	DB30
S1	008	2700	164	25	DB30
S1	009	3000	10	25	DB30
S1	010	5000	347	25	DB30
S1	011	5400	112	25	DB30
S1	012	9000	81	25	DB30

ผลลัพธ์ที่ได้ ของตัวอย่าง ข.8

DIAMETER 12 DB30					
quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*1.00	3*2.60	1*3.00		5	12.0
S1 003 6	S1 002 18	S1 001 6			
4*3.00				5	12.0
S1 001 20					
1*3.00				1	10.0
S1 001 1					
REST STEEL = 8.20					
USE 10.00 M. = 1					
USE 12.00 M. = 11					

ผลลัพธ์ที่ได้ ของตัวอย่าง ข.9

DIAMETER 16 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
5*2.40				23	12.0
S1 004 115					
3*4.00				58	12.0
S1 006 174					
2*5.00				23	12.0
S1 005 16					
3*2.40	1*4.00			1	12.0
S1 004 3	S1 006 1				
1*2.40	1*4.00			1	10.0
S1 004 1	S1 006 1				
REST STEEL = 4.40					
USE 10.00 M. = 1					
USE 12.00 M. = 3					

ผลลัพธ์ที่ได้ ของตัวอย่าง ข.10

DIAMETER 20 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
6*2.00				3	12.0
S1 007 18					
3*2.00				1	10.0
S1 007 3					
REST STEEL = 4.00					
USE 10.00 M. = 1					
USE 12.00 M. = 3					

ผลลัพธ์ที่ได้ ของตัวอย่าง ข.11

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
4*2.70				23	12.0
S1 008 92					
1*3.00	1*9.00			10	12.0
S1 009 10	S1 012 10				
2*5.00				173	10.0
S1 010 346					
2*5.40				56	12.0
S1 011 112					
1*2.70	1*9.00			71	12.0
S1 008 71	S1 012 71				
1*2.70	1*5.00			1	10.0
S1 008 1	S1 010 1				
REST STEEL = 118.40					
USE 10.00 M. = 174					
USE 12.00 M. = 160					

ตัวอย่างที่ ข.12

ข้อมูล

```
C1      001      2320   100   9   RB24
C1      002      1860   100   9   RB24
C1      003      2920    25   9   RB24
C1      004       570    50   9   RB24
C1      005      1070    50   9   RB24
```

ผลลัพธ์ที่ได้

DIAMETER 9 RB24

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*0.57	1*1.07	1*1.86	2*2.32	16	10.0
C1 004 16	C1 005 16	C1 002 32	C1 001 32		
1*0.57	2*1.07	4*2.32		8	12.0
C1 004 8	C1 005 16	C1 001 32			
1*0.57	1*1.07	4*1.86	1*2.92	10	12.0
C1 004 10	C1 005 10	C1 002 40	C1 003 10		
2*1.07	4*1.86	1*2.32		3	12.0
C1 005 6	C1 002 12	C1 001 3			
1*0.57	1*1.86	2*2.32	1*2.92	1	12.0
C1 004 14	C1 002 14	C1 001 28	C1 003 14		
5*2.32				1	12.0
C1 001 5					
1*0.57	2*1.07	2*1.86	1*2.92	1	10.0
C1 004 1	C1 005 2	C1 002 2	C1 003 1		
1*0.57				1	10.0
C1 004 1					
REST STEEL = 11.00					
USE 10.00 M. = 32					
USE 12.00 M. = 22					

ตัวอย่างที่ ข.13

ข้อมูล

C1	002	10000	36	25	RB24
C1	003	11000	32	25	RB24
C1	004	6000	144	25	RB24
C1	005	4650	36	25	RB24
C1	006	3650	36	25	RB24

ผลลัพธ์ที่ได้

DIAMETER 25 RB24

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*3.65	1*4.65			18	12.0
C1 006 36	C1 005 18				
2*4.65				9	10.0
C1 005 18					
2*6.00				72	12.0
C1 004 144					
1*10.00				36	10.0
C1 002 36					
1*11.00				32	12.0
C1 003 32					
REST STEEL = 39.20					
USE 10.00 M. = 45					
USE 12.00 M. = 122					

ตัวอย่างที่ ข.14

ข้อมูล

C1	001	8500	6	12	DB30
C1	002	9000	6	12	DB30
C1	003	10000	42	12	DB30
C1	004	9500	42	12	DB30
C1	005	6000	176	12	DB30

ผลลัพธ์ที่ได้

DIAMETER 12 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*6.00				88	12.0
C1 005 176					
1*8.50				6	10.0
C1 001 6					
1*9.00				6	10.0
C1 002 6					
1*9.50				42	10.0
C1 004 42					
1*10.00				42	10.0
C1 003 42					
REST STEEL = 36.00					
USE 10.00 M. = 88					
USE 12.00 M. = 96					

ตัวอย่างที่ ข.15

ข้อมูล

C1	001	2730	10	25	DB30
C1	002	3230	10	25	DB30
C1	003	3330	70	25	DB30
C1	004	3870	40	25	DB30
C1	005	3300	10	25	DB30
C1	006	3800	10	25	DB30
C1	007	3200	20	25	DB30
C1	008	3700	20	25	DB30
C1	009	3830	10	25	DB30
C1	010	2830	20	25	DB30

ผลลัพธ์ที่ได้

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*2.73	1*2.83	1*3.70		5	12.0
C1 001 10	C1 010 5	C1 008 20			
1*2.83	1*3.33	1*3.83		5	10.0
C1 010 5	C1 003 5	C1 009 5			
3*3.20				6	10.0
C1 007 18					
1*2.83	1*3.23	1*3.87		10	10.0
C1 010 10	C1 002 10	C1 004 10			
3*3.30				3	10.0
C1 005 9					
3*3.33				21	10.0
C1 003 63					

DIAMETER 25 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
3*3.70				5	12.0
C1 008 15					
3*3.80				3	12.0
C1 006 9					
3*3.83				1	12.0
C1 009 3					
3*3.87				10	12.0
C1 004 30					
1*3.80	2*3.83			1	12.0
C1 006 1	C1 009 2				
1*3.20	1*3.33			1	10.0
C1 007 1	C1 003 2				
1*3.20	1*3.30			1	10.0
C1 007 1	C1 005 1				
REST STEEL = 18.60					
USE 10.00 M. = 47					
USE 12.00 M. = 25					

ตัวอย่างที่ ข.16

ข้อมูล

C1	001	3500	14	28	DB30
C1	002	4000	14	28	DB30
C1	003	3550	20	28	DB30
C1	004	4050	20	28	DB30
C1	005	3600	14	28	DB30
C1	006	4100	14	28	DB30
C1	007	3200	134	28	DB30

C1	008	3700	134	28	DB30
C1	009	3680	40	28	DB30
C1	010	4180	40	28	DB30
C1	011	3300	20	28	DB30
C1	012	3800	20	28	DB30
C1	013	2900	40	28	DB30
C1	014	3400	40	28	DB30

ผลลัพธ์ที่ได้

DIAMETER 28 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*2.90	1*3.40	1*3.70		20	10.0
C1 013 20	C1 014 20	C1 008 20			
3*3.20				11	10.0
C1 007 33					
1*2.90	1*3.30	1*3.70		20	10.0
C1 013 20	C1 011 20	C1 008 20			
1*3.20	1*3.40			10	10.0
C1 007 10	C1 014 20				
2*3.20	1*3.50			14	10.0
C1 007 28	C1 001 14				
2*3.20	1*3.55			17	10.0
C1 007 34	C1 003 17				
2*3.20	1*3.60			14	10.0
C1 007 28	C1 005 14				
3*3.68				13	12.0
C1 009 39					
3*3.70				14	12.0
C1 008 42					

DIAMETER 28 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
1*3.70	1*3.80	1*4.18		20	12.0
C1 008 20	C1 012 20	C1 010 20			
1*3.70	1*4.00	1*4.18		14	10.0
C1 008 14	C1 002 14	C1 010 14			
1*3.70	2*4.05			10	12.0
C1 008 10	C1 004 20				
1*3.70	2*4.10			7	12.0
C1 008 7	C1 006 14				
1*3.55	2*4.18			3	12.0
C1 003 3	C1 010 6				
1*3.20	1*3.68	1*3.70		1	12.0
C1 007 1	C1 009 1	C1 008 1			
REST STEEL = 46.20					
USE 10.00 M. = 106					
USE 12.00 M. = 82					

ตัวอย่างที่ ข.17

ข้อมูล

C1	001	2200	150	16	DB30
C1	002	3750	80	16	DB30
C1	003	4000	120	16	DB30
C1	004	1900	250	16	DB30
C1	005	9500	80	16	DB30
C1	006	6400	305	16	DB30
C1	007	7500	95	16	DB30
C1	008	4600	68	16	DB30
C1	009	10000	140	16	DB30

ผลลัพธ์ที่ได้

ผลลัพธ์ที่ได้

DIAMETER 16 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
5*1.90				31	10.0
C1 007 33					
2*2.20	1*6.40			75	12.0
C1 001 150	C1 006 75				
3*3.75				12	12.0
C1 002 36					
1*4.00	1*6.40			120	12.0
C1 003 120	C1 006 120				
1*4.60	1*6.40			68	12.0
C1 008 68	C1 006 68				
1*3.75	1*6.40			42	12.0
C1 002 42	C1 006 42				
1*1.90	1*7.50			95	10.0
C1 004 95	C1 007 95				
1*9.50				80	10.0
C1 005 80					
1*10.00				140	10.0
C1 009 140					
2*3.75				1	10.0
C1 002 2					
REST STEEL = 551.70					
USE 10.00 M. = 347					
USE 12.00 M. = 317					

ตัวอย่างที่ ข.18

ข้อมูล

B1	001	9100	50	16	DB30
B1	002	8200	60	16	DB30
B1	003	5500	50	16	DB30
B1	004	2700	100	16	DB30
B1	005	1400	80	16	DB30
B1	006	1200	100	16	DB30

ผลลัพธ์ที่ได้

DIAMETER 16 DB30

quan.*length	quan.*length	quan.*length	quan.*length	no.cut	st
2*1.20	1*1.40	1*8.20		26	12.0
B1 006 52	B1 005 26	C1 002 26			
2*1.20	1*1.40	1*2.70	1*5.50	20	12.0
B1 006 10	B1 005 20	C1 004 20	B1 003 20		
1*1.20	4*2.70			7	12.0
B1 006 7	B1 004 28				
2*5.50				15	12.0
B1 003 36					
1*1.40	1*8.20			33	10.0
B1 005 33	B1 002 33				
1*2.70	1*9.10			50	12.0
B1 004 50	B1 001 50				
1*2.70	1*8.20			1	12.0
B1 004 1	B1 002 1				
REST STEEL = 44.00					
USE 10.00 M. = 34					
USE 12.00 M. = 119					

จากข้อมูลตัวอย่างที่แสดงนี้ รวมกับข้อมูลทดสอบและตัวอย่างใน บทที่ 5 สามารถนำมาสรุปเป็นตารางได้ดังนี้

ตารางที่ ข.1 ตารางแสดงผลของโปรแกรม YTP ; กับข้อมูลเปรียบเทียบ

ชื่อข้อมูลที่นำมาทดสอบเปรียบเทียบ	YTP			ข้อมูลทดสอบ			ความต่าง		
	เศษ	รวม	%	เศษ	รวม	%	เศษ	รวม	%
นายสันติ									
1) ข้อมูล ชุด2	53.6	13361	0.4	167	16087	6.80	113	2756	6.40
2) ดย. ข.1	61	2396	2.56	335	2792	12.0	274	396	9.44
3) ดย. ข.2	35	10839	0.32	1941	12940	15.0	1906	2101	14.7
ดิกเมืองไทย									
4) ข้อมูล ชุด4	490	24809	1.97	1013	25319	4	523	510	2
5) ดย. ข.3	89.1	25880	0.34	667	26497	2.5	578	617	2.2
6) ดย. ข.4	482	77286	0.62	575	77407	0.74	93	121	0.12
7) ดย. ข.5	3290	72903	4.5	3507	73304	4.8	184	401	0.3
8) ดย. ข.6	7.8	899	0.87	35	959	4	27.2	60	3.13
9) ดย. ข.7	3.3	1479	0.22	112	1524	7.36	109	45	7.14
10)ดย. ข.8	7.3	128	5.7	7.3	128	5.7	0	0	0
11)ดย. ข.9	7	1697	0.4	10	1721	0.6	3	24	0.2
12)ดย. ข.10	456	14091	3.2	741	14538	5.1	2	447	1.1
13)ดย. ข.11	10	114	8.8	15	119	12.5	5	5	3.8
14)ดย. ข.12	25.6	305	1.9	5.5	291	8.4	14	20	6.5
15)ดย. ข.13	151	7369	2	284	8978	3	133	1609	1
16)ดย. ข.14	32	1790	1.7	32	1790	1.7	0	0	0
17)ดย. ข.15	72	2965	0.24	454	3234	14	382	269	13.7
18)ดย. ข.16	223	9872	2.26	2258	11428	22.4	2035	1546	20.1
LOW RISE C.									
19)ข้อมูล ชุด3	109	40564	0.27	905	41125	2.2	736	561	1.93
20)ดย. ข.17	872	11493	7.6	952	11613	8.1	80	120	0.5
21)ดย. ข.18	69.5	2793	2.5	144	3015	4.8	74.5	222	2.3
รวม	6546	323.0 X1000	2.03	14160	334.8 X1000	4.23			2.20

หน่วยทั้งหมดเป็น กิโลกรัม

ข้อมูลชุดที่ 2-4 เป็นข้อมูลที่แสดงผลไว้แล้วใน บทที่ 5 สำหรับข้อมูลทดสอบของ นายสันติ ชินานุวัติวงศ์ เป็นการนำข้อมูลทดสอบในวิทยานิพนธ์ของ นายสันติ ชินานุวัติวงศ์ ที่เก็บจากข้อมูลจริงในหน่วยงานในรูปเปอร์เซนต์การสูญเสียมาคำนวณหาน้ำหนักเหล็กรวม ส่วนข้อมูลอื่น ๆ เป็นข้อมูลจากการทำงานในหน่วยงานจริง

จากตารางสรุปผลการทดสอบข้อมูล แสดงให้เห็นว่า ผลลัพธ์ที่ได้จากโปรแกรม YTP มีเปอร์เซนต์เศษเหล็ก 2.03 % เมื่อใช้กับข้อมูลเปรียบเทียบที่มีเปอร์เซนต์เศษเหล็ก 4.23 % และให้ผลลัพธ์ที่ประหยัดกว่าข้อมูลเปรียบเทียบ 2.20 % ถ้าพิจารณาเปรียบเทียบกับข้อมูลอาคารที่จอดรถตึกเมืองไทยประกันชีวิต ซึ่งมีการเก็บข้อมูลที่ดี โดยใช้ข้อมูลพื้นชั้น 1 ชั้น 2 และเสาบางส่วน พบว่าต้องใช้เหล็กเส้นทั้งหมด 247299 กิโลกรัม แต่เมื่อคำนวณโดยโปรแกรม YTP ต้องใช้เหล็กเส้นทั้งหมด 241587 กิโลกรัม ลดลงกว่าที่ทำการคำนวณการตัดเหล็กด้วยคนถึง 5712 กิโลกรัม คิดเป็นเงิน 5712 X 13.5 บาท/กก. = 77112 บาท ถ้าหากคิดเทียบกับทั้งโครงการซึ่งใช้เหล็กเส้นจากการทำงานจริงถึงหนึ่งพันห้าร้อยตัน แล้ว พบว่า โปรแกรม YTP สามารถช่วยประหยัดเงินได้ถึงเกือบครึ่งล้านบาท

เพื่อความเข้าใจที่ดีขึ้น ตารางที่ ข.2 แสดงถึงการคำนวณการตัดเหล็ก และรูปที่ ข.1 แสดงแปลนพื้นชั้น 2 ของอาคารที่จอดรถตึกเมืองไทยประกันชีวิต ซึ่งได้ถอดแบบข้อมูลออกมาแสดงการคำนวณโดยโปรแกรม YTP ดังแสดงผลใน ตัวอย่าง ข.5

BAR CUTTING LIST

MUANGTHAI PARKING

TOP BAR

1	55	55	2.00	Ø25		9								1	2	
2	55	55	2.70	Ø25		14					1.20 3.90	13 1				
3	20	20	4.00	Ø25		6	2	เสาที่ 3 4.30	2		0.30	2		3	4	
4	20	20	4.30	Ø25			20	เสาที่ 8 4.30	20							
5	1434	1434	5.00	Ø25	718	2								5	6	
6	605	605	5.40	Ø25		302					1.20	302				
7	10	10	5.50	Ø25		10					1.00	10		7	8	
8	22	22	5.70	Ø25	22			เสาที่ 4 4.30	22							
								เสาที่ 3 4.30	2							
9	80	80	6.00	Ø25		40								9	10	
10	324	324	8.40	Ø25		324					3.60	324		วงกบหน้า RAMP & LIFT.		
11	319	319	10.00	Ø25	319									11	12	
12																
13														13	14	
14																
15														15	16	
16																

Ø25 x 10 = 1058 = 13,380

x 12 = 705 = 8460

= 19,040

เกิน = 1.20 x 13 = 15.6

3.90 x 1 = 3.9

Ø3 x 2 = 0.40

1.20 x 302 = 362.4

1 x 10 = 10

* 1.60 x 324 = 518.4

910.90

* Loss = $\frac{910.90}{19040} = 4.78\%$

ต้นฉบับ หน้าขาดหาย

ภาคผนวก ค

โปรแกรม

```
program pulldownmenu;
uses  tet,dos,crt,screen,keyboard,win,ouu,tev,ter,va1,printer;
const maxchoice = 5;
maxmenu         =  4;
type string8    = string [9];
     string80   = string [80];
     fiv        = text;
     structuremenu = record
                       win    : array [1..4]       of byte;
                       col    : array [0..maxchoice] of byte;
                       row    : array [0..maxchoice] of byte;
                       msg    : array [0..maxchoice] of string80;
                       lastchoice : byte;
                     end;
     strucmenu = array [1..maxmenu] of structuremenu;
     functionmenu= record
                     col      :array[1..4] of byte;
                     row      :array[1..4] of byte;
                     msg      :array[1..4] of string8;
                   end;
const menu : strucmenu= (
                         (win:(01,02,17,05);
                     col :(01,02,02,00,00,00);
                     row :(01,01,02,00,00,00);
                     msg :('CALCULATION',
                           'RUN NOT PRINT',
                           'RUN & PRINT',
                           '','','');
                     lastchoice:2),

                     (win:(19,02,33,04);
                      col:(19,02,02,00,00,00);
                      row:(01,01,02,00,00,00);
```

```
                    msg :('FILES'     ,
                          'DATA FILE',
                          '','','','');
                    lastchoice:1),

                    (win:(39,02,69,05);
                     col:(39,02,02,00,00,00);
                     row:(01,01,02,00,00,00);
                     msg :('STEEL LENGTH'          ,
                          '1 ST STANDARD STEEL = ',
                          '2 ST STANDARD STEEL = ',
                          '','','');
                    lastchoice:2),

                    (win:(59,02,74,07);
                     col:(59,02,02,02,02,00);
                     row:(01,01,02,03,04,00);
                     msg :('VARIABLE'     ,
                          'STOP BY STEP',
                          'SET RND',
                          'SHOW   CAL.',
                          'SAVE VAR',
                          '');
                    lastchoice:4)
                    );
fkey    :functionmenu  = (col:(01,25,48,71);
                                  row:(25,25,25,25);
                                  msg:('F1-help',
                                       'F2-rename',
                                       'F3-del',
                                       'F10-quit')
                                  );
const choicemenu  : array [1..maxmenu] of 1..maxchoice =(1,1,1,1);
```

```
      statusmenu  : 1..maxmenu = 1;
      finish      : boolean    = false; { run flag}
var   {key         : char;}               { keep keyboard code}
      fi,bfi      : fiv;
      i:integer;

function made(var fi:fiv) :boolean;
begin
  {$I-} reset(fi);{$I+}
  made:=(IOresult=0);
end;
Procedure  chkPrn;
var  i:integer;
     Regs : Registers;
Begin
 i:=0;
 windowopen(4,4,31,7);
              writeln('   PRINTER  IS NOT OPEN');
                write('     OR OUT OF PAPER      ');
  repeat
       {$I-} write(lst,' '); {$I+}inc(i);
       IF ioresult<>0 then
       begin writeln('   PRINTER IS NOT READY');
               write(' press any key when ready');
             readln; end;
       {$I-} write(lst,' '); {$I+}
  until (ioresult=0) or (i=20);
  windowclose;
end;
procedure ref(var ah:hh);
var i:byte;
    fi:file of real;
begin
```

```
  assign(fi,'ha.yt');
  reset(fi);
  for i:=1 to 15 do
   read(fi,ah[i]);
  close(fi);
end;
procedure saf(var ah:hh);
var i:byte;
    fi:file of real;
begin
  for i:=9 to 16 do ah[i]:=0;
  assign(fi,'ha.yt');
  rewrite(fi);
  for i:=1 to  16 do write(fi,ah[i]);
  close(fi);
end;
procedure displayfunckey;
var i :byte;
begin
   window (1,1,80,25);
   with fkey do begin
      for i := 1 to 4 do begin
        gotoxy (col[i],row[i]);
        setattr(lowdisplay);
        setattr (reverselow);
        write (msg[i]);
       end;
    end;
end;

procedure displaymainmenu(var ah:hh);
var i:byte;
begin
```

```
    window (1,1,80,25);
    setattr (lowdisplay);
    for i :=1 to maxmenu do
      with menu [i] do
      begin
           gotoxy (col[0],row[0]);
           write (msg[0]);
      end;
end;

procedure choiceactive (old,new:byte);
begin
  with menu[statusmenu] do
  begin
  setattr (lowdisplay);
  gotoxy (col[old],row[old]);
  write (msg[old]);
  setattr (reverselow);
  gotoxy(col [new],row[new]);
  write (msg[new]);
  end;
end;

procedure menuactive(new:byte;var ah:hh);
var i:byte;
begin
     setattr(reverselow);
     with menu[new] do
     begin
         gotoxy(col[0],row[0]);
         write (msg [0]);
         setwinattr(lowdisplay);
         setboxattr(lowdisplay);
```

```
          setcharattr(lowdisplay);
          setboxstyle(single);
          setwinheader ('');
          windowopen(win[1],win[2],win[3],win[4]);
          for i:=1 to lastchoice do
          begin
             gotoxy(col[i],row[i]);
             write (msg[i]);
             if (new=3) and (i=1) then
              begin gotoxy(24,1);write(ah[1]:3:1);
                    gotoxy(24,2);write(ah[2]:3:1);end;
           end;
         end;
      choiceactive(1,choicemenu[new]);
end;
procedure moveup;
var currentchoice :byte;
begin
    currentchoice :=choicemenu[statusmenu];
    if currentchoice = 1 then
       choicemenu[statusmenu] :=menu[statusmenu].lastchoice
    else
       choicemenu[statusmenu] :=currentchoice-1;
    choiceactive(currentchoice,choicemenu[statusmenu]);
end;
procedure movedown;
var currentchoice :byte;
begin
    currentchoice :=choicemenu[statusmenu];
    if currentchoice = menu [statusmenu].lastchoice then
       choicemenu[statusmenu]:= 1
    else
       choicemenu[statusmenu] :=currentchoice+1;
```

```
    choiceactive(currentchoice,choicemenu[statusmenu]);
end;
procedure moveforward(var ah:hh);
begin
    windowclose;
    if statusmenu+1 >maxmenu then
       statusmenu:=1
    else
       statusmenu:=statusmenu+1;
    menuactive(statusmenu,ah);
end;
procedure moveback(var ah:hh);
begin
    windowclose;
    if statusmenu = 1 then
       statusmenu:= maxmenu
    else
       statusmenu:=statusmenu-1;
    menuactive(statusmenu,ah);
end;
procedure movetofirstchoice;
begin
    choiceactive (choicemenu[statusmenu],1);
    choicemenu[statusmenu]:=1;
end;
procedure movetolastchoice;
begin
  with menu[statusmenu] do begin
    choiceactive (choicemenu[statusmenu],lastchoice);
    choicemenu[statusmenu]:=lastchoice
  end;
end;
procedure movetofirstmenu(var ah:hh);
```

```
begin
    windowclose;
    statusmenu :=1;
    menuactive(1,ah);
end;
procedure movetolastmenu(var ah:hh);
begin
    windowclose;
    statusmenu :=maxmenu;
    menuactive(maxmenu,ah);
end;
procedure processmenu (num : byte;var ah:hh);
var ch:char;
begin
    setwinattr(reverselow);
    setboxattr(reverselow);
    setcharattr(reverselow);
    setboxstyle(single);
    windowopen (20,09,50,14);
    gotoxy (08,2);
    write ('process menu ',num);
    gotoxy (08,3);
    write ('options number '+chr(choicemenu[statusmenu]+48));
    ch := readkey;
    windowclose;
end;
procedure domenu1(var ah:hh);
begin
  if chr(choicemenu[statusmenu]+48) = '1' then
   begin ah[7]:=2;data(ah,av);
         windowclose; end
  else
   if chr(choicemenu[statusmenu]+48) ='2' then
```

```
    begin chkprn;ah[7]:=1;data(ah,av);
          ah[7]:=2;windowclose; end
end;
procedure domenu2;
begin
  mak(av);
end;
procedure domenu3(var ah:hh);
var d:string[8];
    c:integer;
    currentchoice :byte;
begin
  if chr(choicemenu[statusmenu]+48) ='1' then
   begin windowopen(20,09,62,12);gotoxy (02,1);
         write('FIRST STANDARD STEEL LENGTH ');rer(ah[1]);
         while (ah[1]<0) or (ah[1]>99) do
         begin str(ah[1],d);gotoxy(wherex-length(d),wherey);
               rer(ah[1]);end;windowclose       end
  else if chr(choicemenu[statusmenu]+48) ='2' then
   begin windowopen(20,09,62,12);gotoxy (02,1);
         write('SECOND STANDARD STEEL LENGTH ');rer(ah[2]);
         while (ah[2]<0) or (ah[2]>99) do
         begin str(ah[2],d);gotoxy(wherex-length(d),wherey);
               rer(ah[2]);end;windowclose       end
end;
procedure domenu4(var ah:hh);
var currentchoice,xa,ya :byte;
    d:string[4];
    ch:char;
    c,res:integer;
begin
  if chr(choicemenu[statusmenu]+48) ='1' then
   begin windowopen(20,09,55,12);gotoxy (02,1);
```

```
        write('STOP EVERY STEP OF SOL 0/1/2 : ',ah[3]:1:0);
        gotoxy(wherex-1,wherey);d:=readkey;
        if d='0'then ah[3]:=0 else if d='2'then ah[3]:=2
        else ah[3]:=1;windowclose  end
  else if chr(choicemenu[statusmenu]+48) ='2' then
   begin windowopen(20,09,55,12);gotoxy (02,1);
        write('GET VALUE OF UPPER REBOUND ');xa:=wherex;
        ya:=wherey;
        write(ah[4]:2:2);ch:=readkey;
        if ch<>#13 then
         begin gotoxy(xa,ya);write('    ');
               gotoxy(xa,ya);tet1(ah[4]);end;
        if (ah[4]<0) or (ah[4]>0.99) then ah[4]:=0;
        windowclose end
  else if chr(choicemenu[statusmenu]+48) ='3' then
   begin windowopen(20,09,50,12);gotoxy (02,1);
        if ah[5]=1 then d:='Y' else d:='N';
        write('SHOW CALCULATION SHEET ',d);
        gotoxy(wherex-1,wherey);d:=readkey;
        if (d='Y')or(d='y') then ah[5]:=1 else ah[5]:=2;
           windowclose  end
  else if chr(choicemenu[statusmenu]+48) ='4' then
   begin windowopen(20,09,45,12);gotoxy (02,1);
        if ah[6]=1 then d:='Y' else d:='N';
        write('SAVE VARIABLE ',d);
        gotoxy(wherex-1,wherey);d:=readkey;windowclose;
        if (d='Y')or(d='y') then begin ah[6]:=1;saf(ah)end
        else ah[6]:=2;end
end;

procedure domenu(var ah:hh);
begin
    case statusmenu of
```

```
      1 : domenu1(ah);
      2 : domenu2;
      3 : domenu3(ah);
      4 : domenu4(ah);
    end;
end;
procedure processfunction (num : byte);
var ch:char;
begin
    setwinattr(reverselow);
    setboxattr(reverselow);
    setcharattr(reverselow);
    setboxstyle(single);
    windowopen (25,12,55,14);
    gotoxy (08,2);
    write ('process function ',num);
    ch :=readkey;
    windowclose;windowclose;
end;
procedure dof1;
var ch:char;
begin
  setwinattr(reverselow);
  setboxattr(reverselow);
  setcharattr(reverselow);
  setboxstyle(mix1);
  setwinheader('press any key to exit');
  windowopen (5,4,72,19);writeln;
  writeln('    This program is designed for everyone who would
           like to use.');
  writeln('It is easy to run and use. Almost function can watch
           as you run.');
  writeln('If you do not to use it, please watch the manul.
```

```
          It has any ');
  writeln('function dose not to show in the face of program
          these are:');
  writeln('              FUNCTION FOR DATA FILE');
  writeln('              F 5  :  DELETE  BLOCK');
  writeln('              F 6  :  COPY  BLOCK');
  writeln('              F 7  :  BLOCK  BEGIN');
  writeln('              F 8  :  BLOCK  END');writeln;
  writeln('     If  you have other question plase contact ');
  writeln('         Mr. Yotin  Treratanapan');
  ch:=readkey;setcharattr(lowdisplay);setwinheader('');
  setboxattr(lowdisplay);setwinattr(lowdisplay);windowclose;
end;
procedure dof2(var fi,bfi:fiv);
var aa,ca:string[12];
    ch:char;
begin
  windowopen(20,09,54,13);gotoxy (02,1);
  write('ENTER OLD FILE NAME : ');readln(aa);
  write('       NEW FILE NAME : ');readln(ca);
  if aa='' then begin write('df');aa:='aa';end;
  assign(fi,aa);
  if made(fi) then
   begin assign(bfi,ca);rename(fi,ca);end
  else begin clrscr;writeln('FILE   NOT   FOUND':26);
            writeln('press any key':24);ch:=readkey;end;
  windowclose;
end;

procedure dof3(var fi:fiv);
var a:string[12];
    ch:char;
begin
```

```
  windowopen(20,09,60,13);gotoxy (02,1);
  write('ENTER FILE NAME TO ERASE : ');readln(a);
  if a='' then a:='aa';
  assign(fi,a);
  if made(fi) then erase(fi)
  else begin clrscr;writeln('FILE   NOT-   FOUND':29);
            writeln('press any key':27);ch:=readkey;end;
  windowclose;
end;
procedure quit;
begin
    setattr (highdisplay);
    windowclose;
    cursoron;
    clrscr;
    finish :=true;
end;
procedure testkey (key:char;var ah:hh;var fi,afi:fiv;
                   var funckey:boolean);
begin
    if funckey then
       case  (key) of
             home]key  : movetofirstmenu(ah);
             end]key   : movetofirstmenu(ah);
             up]key    : moveup;
             lt]key    : moveback(ah);
             rt]key    : moveforward(ah);
             dn]key    : movedown;
             pgup]key  : movetofirstchoice;
             pgdn]key  : movetolastchoice;
             f1]key    : dof1;
             f2]key    : dof2(fi,bfi);
             f3]key    : dof3(fi);
```

```
            f10]key    : quit;
end { of case }
else
     if key = return]key then domenu(ah);
end;
begin
    clrscr;ref(ah);
    directvideo :=true;
    checksnow   :=true;
    displayfunckey;
    displaymainmenu(ah);
    menuactive(1,ah);
    cursoroff;i:=0;
    repeat
       readfunckey (key,funckey);
       testkey (key,ah,fi,bfi,funckey);
    until finish;
end.
```

```
unit va1;

interface
uses crt;
type add = array[0..60] of real;
     aff = array[0..5] of real;
     aft = array[0..5] of integer;
     hh  = array[1..16] of real;
var a,c:add;
    af:aff;
    fa:aft;
    ah:hh;
    ha,hb:real;
    key  : char;
    funckey:boolean;
implementation

begin
end.
```

```
unit vav;
interface
type abb = array[1..51,1..20] of real;
var  ab:abb;
implementation

begin
end.
```

```
unit tev;
interface

uses crt,turbo3,dos,screen,win,ter,va1,keyboard;

procedure mak(var av:nm);

implementation

procedure mak(var av:nm);
type fiv=text;
     cw=string[12];
var fi:fiv;
    aw:cw;
    key:char;
    st,sb:string;
    x,y,xa,ya:byte;
    i,c,pta,ptb,ptr:integer;
    quit,fun:boolean;
function made(var fi : fiv) : boolean;
begin
  {$I-} reset(fi); {$I+}
  made:=IOresult = 0;
end;
procedure pttr(var sb:string;ptr:integer;var key:char);
var pr:integer;
begin
  pr:=ptr mod 6;
  case pr of
   1 : if length(sb)<6 then sb:=concat(sb,key);
   2 : if (length(sb)<6) and(key>#47) and(key<#58) then
          sb:=concat(sb,key);
   3 : if (length(sb)<6) and(key>#47) and(key<#58) then
```

```
            sb:=concat(sb,key);
    4 : if (length(sb)<5) and(key>#47) and(key<#58) then
            sb:=concat(sb,key);
    5 : if (length(sb)<3) and(key>#47) and(key<#58) then
            sb:=concat(sb,key);
    0 : if length(sb)<5 then sb:=concat(sb,key);
  end;
  if (key=#8 ) and (length(sb)>0) then
   begin  delete(sb,length(sb),1);
          gotoxy(wherex-1,wherey);write(' ');
   end;
end;
procedure readfunc(var key:char;var sb:string);
begin
  key:=readkey;
  if key= #0 then
   begin
     fun :=true;
     key     :=readkey;
   end
  else
   begin
     fun := false;
     pttr(sb,ptr,key);write(key);
   end;
end;
procedure mapkey(var key:char);
begin
  if fun then
   case key of
     f1]key   : key := ^p;
     f2]key   : key := ^u;
     home]key : key := ^a;
```

```
      end]key  : key := ^z;
      pgup]key : key := ^h;
      pgdn]key : key := ^i;
      lt]key   : key := ^s;
      rt]key   : key := ^d;
      dn]key   : key := ^t;
      up]key   : key := ^q;
      f5]key   : key := ^g;
      f6]key   : key := ^b;
      f7]key   : key := ^c;
      f8]key   : key := ^e;
      f9]key   : key := ^f
    else
      key := #00;
  end;
end;
procedure sei(var pta,ptb,ptr:integer;var av:nm;var sb:string;
              var xa,ya:byte);
var ef:string[7];
begin
  if sb<>'' then
   begin
     ef:='     ';delete(ef,5-length(sb),length(sb));
     if ptr mod 6 =1 then sb:=concat(sb,ef)
     else sb:=concat(ef,sb);
     delete(av[trunc((ptr-1)/6)+1],7*
           (ptr-6*trunc((ptr-1)/6)-1)+1,5);
     insert(sb,av[trunc((ptr-1)/6)+1],7*
           (ptr-6*trunc((ptr-1)/6)-1)+1);
     sb:='';
   end;
  window(3,3,59,23);gotoxy(xa-5,ya);
  if (trunc((ptr+5)/6)>trunc((pta-1)/6))and
```

```
      (trunc((ptr+5)/6)<trunc((ptb+11)/6)) then
    begin
      setattr(reverselow);
      write(copy(av[trunc((ptr-1)/6)+1],7*
           (ptr-6*trunc((ptr-1)/6)-1)+1,5));
      setattr(lowdisplay);
    end
  else begin gotoxy(wherex,ya);
       write(copy(av[trunc((ptr-1)/6)+1],
            7*(ptr-6*trunc((ptr-1)/6)-1)+1,5));
       end;
end;
procedure sej(var pta,ptb,ptr,i:integer;var av:nm;
              var xa,ya:byte);
begin
  if (trunc((ptr+5)/6)>trunc((pta-1)/6))and
     (trunc((ptr+5)/6)<trunc((ptb+11)/6)) then
   write(copy(av[trunc((ptr-1)/6)+1],
         7*(ptr-6*trunc((ptr-1)/6)-1)+1,5))
  else begin
         setattr(reverselow);
         write(copy(av[trunc((ptr-1)/6)+1],
               7*(ptr-6*trunc((ptr-1)/6)-1)+1,5));
         setattr(lowdisplay);
       end;
  xa:=wherex;ya:=wherey;
  window(1,1,80,25);gotoxy(1,1);clreol;
  {write('pr',ptr,'y',ya,'x',xa,'i',i);}
end;
procedure leftarrow(var sb:string;var ptr,i:integer;
                    var av:nm;var xa,ya:byte);
var j,k:integer;
begin
```

```
  if (ptr>1) and ((ya*xa)>6)  then
   begin
    if ptr mod 6 = 1 then
     begin
       sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr-1;
       gotoxy(wherex+30,wherey-1);sej(pta,ptb,ptr,i,av,xa,ya);
     end
    else
     begin
       sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr-1;
       gotoxy(wherex-12,wherey);sej(pta,ptb,ptr,i,av,xa,ya);
     end;
   end
  else begin
         if ptr+119>i then begin j:=i-ptr;k:=trunc(j/6)+1;end
         else if i<120 then begin j:=i-1;k:=trunc(i/6)end
         else begin j:=119;k:=20;end;
         window(3,3,59,23);
         gotoxy(1,1);writeln(av[trunc((ptr+5)/6)]);ptr:=ptr+j;
         gotoxy(36,k);sej(pta,ptb,ptr,i,av,xa,ya);
       end;
end;
procedure rightarrow(var sb,st:string;var av:nm;
                     var i,ptr:integer;var xa,ya:byte);
var k:integer;
begin
  if (ptr<i) and(ya<21) and(6*ya-frac(ptr/6)<120) then
   begin
     if ptr mod 6 = 0 then
      begin
        sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr+1;
        gotoxy(1,wherey+1);sej(pta,ptb,ptr,i,av,xa,ya);
      end
```

```
      else
       begin
         sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr+1;
         gotoxy(wherex+2,wherey);
         sej(pta,ptb,ptr,i,av,xa,ya);
       end;
    end
  else begin
         sei(pta,ptb,ptr,av,sb,xa,ya);
         if ya<20 then ptr:=ptr-6*ya+1
         else if i<120 then ptr:=1 else ptr:=ptr-119;
         gotoxy(1,1);sej(pta,ptb,ptr,i,av,xa,ya);
       end;
end;
procedure down(var sb:string;var av:nm;var pta,ptb,ptr,i:integer;
               var xa,ya:byte);
var yb:byte;
    sc,sd:string[5];
begin
  if (ptr+6<=i) and (ya<20) then
   begin
     sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr+6;
     gotoxy(wherex-5,wherey+1);sej(pta,ptb,ptr,i,av,xa,ya);
   end
  else
   begin
     if ptr+6>i then i:=i+6;
     window(3,3,59,23);str(trunc((ptr+5)/6)+1,sc);sd:='  000';
     delete(sd,5-length(sc),length(sc));sd:=concat(sd,sc);
     av[trunc((ptr+5)/6)+1]:=av[trunc((ptr-1)/6)+1];
     delete(av[trunc((ptr+5)/6)+1],8,5);
     insert(sd,av[trunc((ptr+5)/6)+1],8);
     if ya=20 then yb:=21 else yb:=ya+1;
```

```
      gotoxy(1,yb);
      if (trunc((ptr+5)/6+1)>trunc((pta-1)/6))and
         (trunc((ptr+5)/6+1)<trunc((ptb+11)/6)) then
        begin setattr(reverselow);write(av[trunc((ptr+5)/6)+1]);
              setattr(lowdisplay);writeln; end
        else writeln(av[trunc((ptr+5)/6)+1]);
      if ya=20 then begin yb:=yb-1;ya:=ya-1 end;
      sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr+6;
      gotoxy(wherex-5,yb);sej(pta,ptb,ptr,i,av,xa,ya);
    end;
end;
procedure up(var sb:string;var av:nm; var i,pta,ptb,ptr:integer;
             var xa,ya:byte);
var k,j:integer;
    xb:byte;
begin
  if ptr-6>0 then
   begin
     sei(pta,ptb,ptr,av,sb,xa,ya);ptr:=ptr-6;
     if wherey=1 then
      begin
        k:=trunc(i/6)-trunc((ptr+5)/6)+1;xb:=wherex;
        if k>19 then k:=20
        else k:=trunc(i/6)-trunc((ptr-1)/6);
        for j:=1 to k do
         begin
           gotoxy(1,wherey);
           if (trunc((ptr-1)/6+j)>trunc((pta-1)/6))and
              (trunc((ptr-1)/6+j)<trunc((ptb+11)/6)) then
            begin setattr(reverselow);
            writeln(av[trunc((ptr-1)/6)+j]);
                  setattr(lowdisplay);end
            else writeln(av[trunc((ptr-1)/6)+j]);
```

```
          end;
         gotoxy(xb-5,1);sej(pta,ptb,ptr,i,av,xa,ya)
       end
     else
      begin gotoxy(wherex-5,wherey-1);
      sej(pta,ptb,ptr,i,av,xa,ya) end
    end;
end;
procedure pgup(var av:nm;var pta,ptb,ptr:integer;var xa,ya:byte);
var j:integer;
begin
  if ptr-120-6*ya>0 then
   begin
     window(3,3,59,23);ptr:=ptr-114-6*ya;gotoxy(1,wherex-ya);
     for j:=1 to 20 do
      if (trunc((ptr-1)/6+j)>trunc((pta-1)/6))and
         (trunc((ptr-1)/6+j)<trunc((ptb+11)/6)) then
       begin setattr(reverselow);writeln(av[trunc((ptr-1)/6)+j]);
            setattr(lowdisplay);end
       else writeln(av[trunc((ptr-1)/6)+j]);
     gotoxy(wherex-5,wherey-1);window(1,1,80,25);gotoxy(1,1);
     ptr:=ptr+6*ya-6;
   end;
end;
procedure pgdn(var av:nm;var pta,ptb,ptr,i:integer;
               var xa,ya:byte);
               var j,k:integer;
begin
  if ptr+240-6*ya<i then
   begin
     window(3,3,59,23);ptr:=ptr+120-6*ya;gotoxy(1,wherex-ya);
     for j:=1 to 20 do
      if (trunc((ptr-1)/6+j)>trunc((pta-1)/6))and
```

```
          (trunc((ptr-1)/6+j)<trunc((ptb+11)/6)) then
        begin setattr(reverselow);
        writeln(av[trunc((ptr-1)/6)+j]);
              setattr(lowdisplay);end
        else writeln(av[trunc((ptr-1)/6)+j]);
      gotoxy(wherex-5,wherey-1);window(1,1,80,25);gotoxy(1,1);
            ptr:=ptr+6*ya-6;
   end;
end;
procedure ctt(var pta,ptb,ptr,i:integer;var av:nm;var ya:byte);
{f8}
var c,k:integer;
begin
  window(3,3,59,23);
  if ptr+6*(20-ya) <i then k:=20
  else k:=trunc((i-ptr+6*ya-1)/6);
  gotoxy(1,1);
  for c:= 1 to k do
   if (ptr-6*ya+6*c>=pta) and(ptr+6*(c-ya)<=ptb) then
    begin
      setattr(reverselow);writeln(av[trunc((ptr-6*ya+5)/6)+c]);
      setattr(lowdisplay);
    end
   else
   writeln(av[trunc((ptr-6*ya+5)/6)+c]);
  gotoxy(xa,ya);
  window(1,1,80,25);
end;
procedure hte(var pta,ptb,ptr:integer;var av:nm;var xa,ya:byte);{f5}
var c,k:integer;
begin
  window(3,3,59,23);
  if ptr+6*(20-ya) <i then k:=20
```

```
  else k:=trunc((i-ptr+6*ya-1)/6);
  gotoxy(1,1);
  for c:= 1 to k do
   writeln(av[trunc((ptr-6*ya+5)/6)+c]);
  gotoxy(xa,ya);pta:=ptb+6;
  window(1,1,80,25);
end;
procedure ett(var pta,ptb,ptr,i:integer;var av:nm;var xa,ya:byte);
{f6}
var xb,yb:byte;
    c,j,k,m:integer;
    sb,ef:string[5];
begin
  xb:=wherex;yb:=wherey;window(3,3,59,23);
  j:=trunc((ptb-1)/6)-trunc((pta-1)/6)+1;
  for c:= trunc((pta-1)/6)+1 to trunc(i/6) do
   begin
     str(c,sb);
     ef:='  000';delete(ef,5-length(sb),length(sb));
     sb:=concat(ef,sb);delete(av[c+j],8,5);
     insert(sb,av[c+j],8);av[c]:=av[c+j];
   end;
  for c:= trunc(i/6)-j+1 to trunc(i/6) do
   av[c]:='';m:=i;i:=i-6*j;clrscr;
  if ptr+6*(20-ya) <i then k:=20
  else k:=trunc((i-ptr+6*ya-1)/6);
  gotoxy(1,1);
  for c:= 1 to k do
   writeln(av[trunc((ptr-6*ya+5)/6)+c]);
  ptr:=ptr-6*ya+6;yb:=ya;ya:=1;pta:=ptb+6;
  window(1,1,80,25);
end;
procedure ftt(var pta,ptb,ptr,i:integer;var av:nm;var xa,ya:byte);
```

```
{f9}
var k,j,c:integer;
    sb,ef:string[5];
    sa:string[45];
begin
  window(3,3,59,23);
  j:=trunc((ptb-1)/6)-trunc((pta-1)/6)+1;
  for k:= trunc(i/6) downto trunc((ptr+5)/6) do
   begin
     str(k+j,sb);ef:='  000';
     delete(ef,5-length(sb),length(sb));delete(av[k],8,5);
     insert(concat(ef,sb),av[k],8);av[k+j]:=av[k];
   end;
  if pta>ptr then c:=j else c:=0;
  for k:=trunc((ptr+5)/6) to trunc((ptr-1)/6)+j do
   begin
     str(k,sb);ef:='  000';inc(c);sa:=av[trunc((pta-1)/6)+c];
     delete(ef,5-length(sb),length(sb));delete(sa,8,5);
     insert(concat(ef,sb),sa,8);av[k]:=sa;
   end;
  i:=i+6*j;ptr:=i;clrscr;
  if pta>ptr then k:=ptr else k:=pta;
  pta:=ptb;
  for k:=trunc((k-1)/6)+1 to trunc(i/6) do
   writeln(av[k]);ya:=wherey-1;xa:=41;pta:=ptb+6;
  window(1,1,80,25);
end;
procedure processfunckey(key:char;var st:string;var av:nm;
                         var xa,ya:byte;
                         var i,pta,ptb,ptr:integer;
                         var quit:boolean);
begin
  case key of
```

```
    ^m : begin if st= '' then st:=#13; quit:=true; end;
    ^[ : begin st:= #27;quit:=true; end;
    ^s : leftarrow(sb,ptr,i,av,xa,ya);
    ^d : rightarrow(sb,st,av,i,ptr,xa,ya);
    ^t : down(sb,av,pta,ptb,ptr,i,xa,ya);
    ^q : up(sb,av,i,pta,ptb,ptr,xa,ya);
    ^h : pgup(av,pta,ptb,ptr,xa,ya);
    ^i : pgdn(av,pta,ptb,ptr,i,xa,ya);
    ^c : begin pta:=ptr;ctt(pta,ptb,ptr,i,av,ya);end;
    ^e : begin ptb:=ptr;ctt(pta,ptb,ptr,i,av,ya);end; {f8Kk}
    ^b : ett(pta,ptb,ptr,i,av,xa,ya);                {f6Ky}
    ^f : ftt(pta,ptb,ptr,i,av,xa,ya);          {f9Kc}
    ^g : hte(pta,ptb,ptr,av,xa,ya);                {f5Kh}
  end
end;
procedure te(var i:integer;var av:nm;var fi:fiv;var aw:cw);
var c:integer;
begin
  write('ENTER FILE"S NAME : ');readln(aw);c:=0;aw:=aw+'.txt';
  clrscr;
  assign(fi,aw);
  if made(fi) then
   begin
     reset(fi);
     while not eof(fi) and (c<50) do
      begin
        c:=c+1;
        readln(fi,av[c]);
        av[c]:=concat(copy(av[c],1,5),'  ',copy(av[c],7,5),'   ',
              copy(av[c],20,5),' ',copy(av[c],26,5),'     ',
              copy(av[c],33,2),'   ',copy(av[c],37,4),' ');
        writeln(av[c]);
      end;
```

```
      close(fi);
    end
  else
   begin av[1]:='B1       001    8000     1      6   RB24';
         c:=1;writeln(av[1]);end;
  i:=c*6;gotoxy(wherex,wherey-1);
end;
procedure tc(var i:integer;var av:nm;var fi:fiv;var aw:cw);
var c:integer;
begin
 { windowopen(12,4,60,15);
   for c:=1 to 4 do
   writeln(av[c]);}
  c:=0;i:=trunc(i/6);
  assign(fi,aw);rewrite(fi);
  while c<i do
   begin
     inc(c);
     delete(av[c],6,1);insert('     ',av[c],13);
     delete(av[c],31,3);delete(av[c],35,1);
     writeln(fi,av[c]);
   end;
   {for c:=1 to 4 do
   writeln(av[c]);
   writeln('12345678901234567890123456789012345678901234567890');readln;}
  close(fi);
end;
begin
  setwinheader('ENETR To Exit');
  windowopen(2,2,60,24);sb:='';clrscr;
  te(i,av,fi,aw);quit:=false;
  st:=key;ptr:=i-5;pta:=0;ptb:=0;
  xa:=wherex;ya:=wherey;
```

```
  while (not(quit)) and (ptr<3300) do
   begin
     readfunc(key,sb);
     mapkey(key);
     processfunckey(key,st,av,xa,ya,i,pta,ptb,ptr,quit);
   end;
  gotoxy(1,1);clreol;setattr(reverselow);write('CALCULATION');
  gotoxy(19,1);write('FILES');
  gotoxy(39,1);write('STEEL LENGTH');
  gotoxy(59,1);write('VARIBLE');setattr(lowdisplay);
  windowclose;
  windowopen(20,11,45,13);write(' SAVE THIS FILE  Y/N ');
  key:=readkey;windowclose;
  if (key='y') or (key='Y') then tc(i,av,fi,aw);
end;
begin
end.
```

```
unit ouu;
interface

uses crt,turbo3,dos,screen,keyboard,vav,va1;
type agd = array[0..850] of real;
var ad:agd;
procedure Dim;
procedure swac(var rb,rc:integer;var ab:abb;var c:add);
procedure nrp(var i,rm:integer;var ab:abb;var c:add);
procedure comb(var fa:aft;var aa:att;var rd,rb,rm:integer;
               ha,hb:real;var ab:abb;
               var a,c:add;var af:aff);

implementation

Procedure Dim;
var   Path    : String;
      DirInfo : SearchRec;
procedure Dir(Mask :String);
begin
  FindFirst(Mask,AnyFile,DirInfo);
  While DosError = 0  do begin
       FindNext (DirInfo);
       if (DosError =0) then Writeln(DirInfo.Name);
  end;
end;
begin
 Path := '*.txt';
 Dir(Path);
end;

procedure swac(var rb,rc:integer;var ab:abb;var c:add);
var mk:real;
```

```
    i,j,k:integer;
begin
  for i:= 1 to rb do
   for j:= i to rb do
    if c[i]>c[j] then
     begin
       mk:=c[j];c[j]:=c[i];c[i]:=mk;
       for k:= 1 to rc do
        begin
          mk:=ab[j,k];ab[j,k]:=ab[i,k];ab[i,k]:=mk;
        end;
     end;
end;
procedure nrp(var i,rm:integer;var ab:abb;var c:add);
var f,l,j,k:integer;
begin
  k:=0;
  for l:= 1 to i-1 do
   if (abs(c[l]-c[i])<0.02) and(k<>rm) then
    begin
      for j:=1 to rm do
       if abs(ab[l,j]-ab[i,j])<0.01 then
        k:=k+1;
      if k=rm then i:=i-1
      else k:=0;
    end;
end;

procedure comb(var fa:aft;var aa:att;var rd,rb,rm:integer;
               ha,hb:real;var ab:abb;var a,c:add;var af:aff);
var he,hf,e,f,h,i,j,k,l,m,n,o,p,q,r,s,t,u,y,v,w,x:integer;
    hc,hd,ht,ct,mk:real;
```

```
procedure htt(h:integer;var ct,ht:real;var af:aff);
var i:integer;
begin
  for i:=0 to h do
   if ct>af[i] then
    ht:=af[i+1];
end;
procedure hta(h:integer;var ct,ht:real;var af:aff;var f:integer);
var z,j:integer;
begin
  j:=0;
  for z:=f to h do
   if (ct<=af[z+1])and (j=0) then
    begin
      f:=z+1;j:=1;{write('h',h,'z',z,'af',af[f]:2:2,'f',f);}
      ht:=af[f];
    end;
end;

begin
  hc:=af[1];hd:=af[2];he:=fa[1];hf:=fa[2];
  for i:=1 to 2 do
   af[i+2]:=af[i];
  af[1]:=ha;af[2]:=hb;
  h:=0;af[0]:=0;{give value of steel}
  for i:= 1 to 4 do
   for j:= i to 4 do
    if af[i]>af[j] then
     begin
       mk:=af[j];af[j]:=af[i];af[i]:=mk;
     end;
  for i:= 1 to 4 do
   if af[i]=0 then inc(h);
```

```
  for i:= 1 to 4 do
   af[i]:=af[i+h];
{   for i:= 1 to 4 do
    write(I:2,' af ',af[i]:2:2,'    ');
    write('h ',h,' ha ',ha:2:2);readln;}
  h:=3-h;
  i:=0;
  for j:= 1 to rm do
   for o:= 0 to aa[j] do
    begin {rm=1}
       if (rm=1) and (o*a[j]<=ha) then
        begin
         ct:=o*a[j];f:=0;
         if i>35 then htt(h,ct,ht,af);
         if (ct>=0)and(o>0)then
          if i<36 then
           while (f<rd+1) and(ct<af[h+1]) do
            begin
              hta(h,ct,ht,af,f);
              i:=i+1;
              ab[i,j]:=o;ab[i,k]:=p;
              c[i]:=ht-ct;
              if i=35 then
               swac(i,rm,ab,c);
            end
          else if (i>35) and (c[35]>(ht-ct)) then
           begin
             ab[i,j]:=o;ab[i,k]:=p;
             c[i]:=ht-ct;
             swac(rb,rm,ab,c);
           end;
        end;
    for k:= j+1 to rm do
```

```
for p:= 0 to aa[k] do
 begin
  if (rm=2) and ((o*a[j]+p*a[k])<=ha) then
   begin
    ct:=o*a[j]+p*a[k];f:=0;
    if i>35 then htt(h,ct,ht,af);
    if (ct>=0)and(o>0)and(p>0)then
     if i<36 then
      while (f<rd+1) and(ct<af[h+1]) do
       begin
         hta(h,ct,ht,af,f);
         i:=i+1;
         ab[i,j]:=o;ab[i,k]:=p;
         c[i]:=ht-ct;
         if i=35 then
          swac(i,rm,ab,c);
       end
     else if (i>35) and (c[35]>(ht-ct)) then
      begin
        ab[i,j]:=o;ab[i,k]:=p;
        c[i]:=ht-ct;
        swac(rb,rm,ab,c);
      end;
   end;
  for l:= k+1 to rm do
   for q:= 0 to aa[l] do
    begin
     if (rm=3) and((o*a[j]+p*a[k]+q*a[l])<=ha)  then
      begin
       ct:=o*a[j]+p*a[k]+q*a[l];f:=0;
       if i>35 then htt(h,ct,ht,af);
       if (ct>=0)and(o+p>0)and(o+q>0)and(q+p>0)then
        if i<36 then
```

```
          while (f<rd+1) and(ct<af[h+1]) do
           begin
            hta(h,ct,ht,af,f);
            i:=i+1;
            ab[i,j]:=o;ab[i,k]:=p;ab[i,l]:=q;
            c[i]:=ht-ct;
            if i=35 then
             swac(i,rm,ab,c);
           end
         else if (i>35) and (c[35]>(ht-ct)) then
          begin
           ab[i,j]:=o;ab[i,k]:=p;ab[i,l]:=q;
           c[i]:=ht-ct;
           swac(rb,rm,ab,c);
          end;
      end;
if rm>3 then
 for m:= l+1 to rm do
  for r:= 0 to aa[m] do
   Begin
   if (o*a[j]+p*a[k]+q*a[l]+r*a[m])<=ha then
    begin
     ct:=o*a[j]+p*a[k]+q*a[l]+r*a[m];f:=0;
     if i>35 then htt(h,ct,ht,af);
     if (ct>0)and(o+p+q>0)and(r+o+q>0)and(r+o+p>0)and(
        r+q+p>0)then
      if i<36 then
       begin
        while (f<rd+1) and(ct<af[h+1])  do
         begin
          hta(h,ct,ht,af,f);
          i:=i+1;
          for t:=1 to rm do
```

```
                    ab[i,t]:=0;
                   ab[i,j]:=o;ab[i,k]:=p;ab[i,l]:=q;ab[i,m]:=r;
                   c[i]:=ht-ct;
                   nrp(i,rm,ab,c);
                   if i=35 then
                     swac(i,rm,ab,c);
                  end;
                end
               else if (i>35) and (c[35]>(ht-ct)) then
                begin
                 for t:=1 to rm do
                  ab[i,t]:=0;
                 ab[i,j]:=o;ab[i,k]:=p;ab[i,l]:=q;ab[i,m]:=r;
                 c[i]:=ht-ct;
                 nrp(i,rm,ab,c);
                 swac(i,rm,ab,c);
                 i:=35;
                end;
             end;
          End;end;End;end;
          rb:=i;af[1]:=hc;af[2]:=hd;fa[1]:=he;fa[2]:=hf;
          af[3]:=0;af[4]:=0;
end;
begin
end.
```

```
unit ter;

interface
uses dos,crt,screen,win,tet,keyboard,printer,ouu,vav,va1;

type nm = array[1..900] of string[44];
var z,pp,kt :  integer;
    av:nm;
procedure rei(var ia:integer);
procedure rer(var ca:real);
procedure ou4(var av:nm;var da:agt;var ad:agd;var z,pp,kt:integer;
              var ha,hb:real;var ah:hh;var af:aff;var fa:aft);
procedure data(var ah:hh;var av:nm);

implementation


procedure rer(var ca:real);
var c:integer;
    b:real;
    xa,ya:byte;
begin
  b:=ca;xa:=wherex;ya:=wherey;
  repeat
   readrealnum(ca,c);
   if c=255 then begin gotoxy(xa,ya);clreol;end
   else if c=1 then ca:=b;
  until (c=0) or(c=1);
end;
procedure rei(var ia:integer);
var c,b:integer;
    xa,ya:byte;
begin
```

```
  b:=ia;ia:=1;xa:=wherex;ya:=wherey;
  repeat
   readintnum(ia,c);
   if c=255 then begin gotoxy(xa,ya);clreol;end
   else if c=1 then ia:=b;
  until c=0;
end;

procedure ou4(var av:nm;var da:agt;var ad:agd;var z,pp,kt:integer;
              var ha,hb:real;var ah:hh;var af:aff;var fa:aft);
var i,j,k,l,m,t,u,v,w,x,y,ra,rb,rd,rm,rn:integer;
    q,ac,bt,ck,ct,ht,mk,wv:real;
    p:string[7];
    ch,chi:char;

function rnd(ip:real;var ah:hh):integer;
var pt:real;
begin
  if ah[4]=0 then pt:=ah[8]
  else pt:=ah[4];
  if frac(ip)>pt+0.002 then rnd:=trunc(ip)+1
  else rnd:=trunc(ip);
end;
procedure  rea(var ac:real);
var x,y:byte;
begin
  tet1(ac);
  if (ac<0) or (ac>12) then
   begin
     x:=wherex+7;y:=wherey-1;
     gotoxy(x,y);write('  ');
     x:=wherex-2;y:=wherey;
     gotoxy(x,y);
```

```
      tet1(ac);
    end;
   x:=wherex+12;y:=wherey-1;
   gotoxy(x,y);
end;
procedure out(rd,rm,rb:integer;var ab:abb;var af:aff);
var i,j:integer;
begin
  write('  No ');
  for i:=1 to rm do
   write(a[i]:3:2,'  ');
  for i:=1 to rd do
   write(af[i]:3:2,'  ');
  writeln(' Rest');
  for i:=1 to rb+rm+1 do
   begin
     write(i:3,'  ');
     for j:= 1 to rm+rd+1 do
      write(ab[i,j]:3:2,'  ');writeln;
     if i = 20 then readln;
   end;writeln(' y ',y);
end;
procedure swaf(rd:integer;var fa:aft;var af:aff);
var mk:real;
    i,j,ml:integer;
begin
 for i:= 1 to rd do
  for j:= i to rd do
   if af[i]>af[j] then
    begin
      mk:=af[j];af[j]:=af[i];af[i]:=mk;
      ml:=fa[j];fa[j]:=fa[i];fa[i]:=ml;
    end;
```

```
end;

procedure sbt(var af:aff;var rb,rm,rd:integer;var ab:abb);
var i,j:integer;
begin
  if rd>0 then
  for i:= 1 to rb do
   for j:= rm+1 to rm+rd do
    if (ab[i,rm+rd+1]=0) and(ab[i,j]=1) then
     ab[i,rm+rd+1]:=ab[i,rm+rd+1]-12+af[j];
end;
procedure csum(var ah:hh);
var j:byte;
begin
  for j:=1 to 4 do
   if ah[8+j]*ah[12+j] <> 0 then
    begin
     if ah[7]=1 then
     writeln(lst,'USE ':20,ah[8+j]:3:2,' M. = ', ah[12+j]:4:0);
     writeln('USE ',ah[8+j]:3:2,' M. = ', ah[12+j]:4:0);
    end;
  if ah[7]=1 then begin writeln(lst,'');writeln(lst,'');end;
  for j:=8 to 15 do ah[j]:=0;
end;

procedure rdf(var fa:aft;var rb,rd,rm:integer;var ab:abb;
var af:aff;var a:add);
var i,j,k,l:integer;
    mk,g:real;
begin
  if rd>0 then
   begin
    ab[rb+rm+1,rm+rd+1]:=0;
```

```
for i:= 1 to rb+rm do
 for j:= rm+1 to rm+rd do
  ab[i,j]:=0;
for i:= 1 to rb do
 begin
   mk:=0;
   for j:=1 to rm do
    mk:=mk+ab[i,j]*a[j];
   if ab[i,rm+rd+1]>=0 then g:=0
   else g:=ab[i,rm+rd+1];
   for k:= 1 to rd do
    begin
     if ((mk+ab[i,rm+rd+1]-g)>af[k]-0.001)and
        ((mk+ab[i,rm+rd+1]-g)<af[k]+0.001) then
      ab[i,rm+k]:=1
     else ab[i,rm+k]:=0;
    end;
 end;
for i:=1 to rb+rm do
 for j:=rm+1 to rm+rd do
  ab[rb+rm+1,j]:=ab[rb+rm+1,j]+ab[i,j];
x:=0;
for k:=rm+1 to rm+rd do
 if ab[rm+rb+1,k]=0 then
  begin
    for j:= k to rm+rd-x do
     begin
       af[j]:=af[j+1];
       fa[j]:=fa[j+1];
       for i:=k to rb+rm do
        ab[i,j]:=ab[i,j+1];
     end;
    x:=x+1;
```

```
      end;
      rd:=rd-x;
      for i:=1 to rd do
       ab[rb+rm+1,rm+i]:=fa[i];
   end;
end;
procedure intg(var ab:abb;var c:add;var aa:att;var ch,chi:char);
var i:integer;
    p,m:string[10];
begin
  ch:='b';c[rm+1]:=0;
  for i:= 1 to rm do
   begin
     str(ab[rb+rm+1,i]:6:2,p);delete(p,1,length(p)-2);
     if p <> '00' then
     ch:='a';
   end;
  if ch='b' then
   for i:= 1 to rm do
    begin
      aa[i]:=sa[i];
      c[i]:=ab[rb+rm+1,i];chi:='b';
      if ae[sa[i],rm+rd+1]>-12 then
       c[rm+1]:=c[rm+1]+(abs(ae[sa[i],rm+rd+1])*ab[rb+rm+1,i]);
    end;
end;

procedure sol(var av:nm;var a:add;
ch,chi:char;var z,pp,kt,rb,rm,rd:integer;
                  var b,aa:att;var ab,ae:abb;var ah:hh);
var i,l,rc:byte;
    k,k1,mc,j,pn:integer;
    cb,pa:real;
```

```
procedure pnn(var pn:integer);
begin
  pn:=wherey+pn;
  if pn>25 then
   begin gotoxy(16,wherey);write(chr(179));
         gotoxy(32,wherey);write(chr(179));
         gotoxy(48,wherey);write(chr(179));
         gotoxy(64,wherey);write(chr(179));
         gotoxy(74,wherey);write(chr(179));gotoxy(1,wherey);end;
end;
procedure jor(var pn,z,pp,k,l:integer;var pa,ak:real;var av:nm;
var ah:hh);
var i :byte;
    res,st,p:integer;
    mt:real;
    dt,et:string[6];
begin
  k:=0;st:=-1;
  repeat
   inc(pp);val(copy(av[pp],20,4),mt,res);mt:=mt/100;
   if (mt=ak) and(copy(av[pp],29,2)<>' 0') then
    begin
      val(copy(av[pp],25,6),st,res);
      st:=st-trunc(pa);
      if st >=0 then
       begin
         if ah[7]=1 then writeln(lst,'':20,
            copy(av[pp],1,4),' ',copy(av[pp],9,3),pa:5:0);
         pnn(pn);
         gotoxy(wherex+16*l,wherey);
         writeln(copy(av[pp],1,4),' ',copy(av[pp],9,3),pa:5:0);
         str(st,dt);delete(av[pp],25,6);et:='      ';
         delete(et,1,length(dt));
```

```
          dt:=concat(et,dt);insert(dt,av[pp],25);
          inc(k);
        end
       else begin
        if ah[7]=1 then writeln(lst,'':20,copy(av[pp],1,4),' ',
           copy(av[pp],9,3),pa:5:0);
        pnn(pn);
        gotoxy(wherex+16*l,wherey);
        writeln(copy(av[pp],1,4),' ',copy(av[pp],9,3),
        copy(av[pp],25,6));
        dt:='     0';delete(av[pp],25,6);insert(dt,av[pp],25);
        inc(k);pa:=-st;  end;
    end;
  until  (pp=z-1) or (pp=100) or (st>=0);
end;
procedure tabl(var z,pp,rm,rb,rd:integer;var av:nm;var ab,ae:abb;
               var sa:att;var a:add;var ah:hh);
var i,k,k1,l,j,m,t:integer;
    ak,pc:real;
    ch:char;
procedure tef(var av:nm;pp:integer);
const  single      = 1;
       double      = 2;
       mix1        = 3;
       mix2        = 4;
       boxstyle         : byte     = single;
       attrofbox:byte = highdisplay;
       attrofwindow:byte = highdisplay;
       attrofheader:byte = highdisplay;
       attrofchar:byte = highdisplay;
       headerofwindow : string ='';

       typeofbox        : array [1..4,1..8] of char =
```

```
                    ((#196,#179,#179,#196,#218,#191,#192,#217),
                     (#205,#186,#186,#205,#201,#187,#200,#188),
                     (#205,#179,#179,#205,#213,#184,#212,#190),
                     (#196,#186,#186,#196,#214,#183,#211,#189));
type  screenline = array[1..80] of integer;
      screenarray= array[1..25] of screenline;
      screenblock= array[1..2000] of integer;
      windowlink = ^windowcontrolblock;
      windowcontrolblock = record
                             x1,y1,x2,y2 : integer;
                             x,y         : integer;
                             id          : byte;
                             backlink       : windowlink;
                             screenconcents : screenblock;
                           end;
var activewindow  :windowlink;
    screenptr     :^screenarray;
    fixedsize     :integer;
    windowcount   :byte;
    x1,y1,x2,y2,x3,y3,x4,y4,x5,x6,x7 : byte;
procedure windowbox(x1,y1,x2,y2,x3,x4,x5,x6,x7,y3:byte);
const top     = 1;left    = 2;
      right   = 3;bottom = 4;
      upleft  = 5;upright  = 6;
      loleft  = 7;loright = 8;
var   x,y :byte;
begin
  window(x1,y1,x2,y2);
  setattr(attrofwindow);
  window(1,1,80,25);
  setattr(attrofbox);
  gotoxy(x1,y1);
  write(typeofbox[boxstyle,upleft]);
```

```
for x := x1+1 to x2-1 do
 write(typeofbox[boxstyle,top]);
write(typeofbox[boxstyle,upright]);
  for x := x1+1 to x2-1 do
   begin
 gotoxy(x,y3);write(typeofbox[boxstyle,top]);end;

for y:= y1+1 to y2-1 do begin
 gotoxy(x1,y);write(typeofbox[boxstyle,left]);
 gotoxy(x2,y);write(typeofbox[boxstyle,right]);
 gotoxy(x3,y);write(typeofbox[boxstyle,left]);
 gotoxy(x4,y);write(typeofbox[boxstyle,right]);
 gotoxy(x5,y);write(typeofbox[boxstyle,left]);
 gotoxy(x6,y);write(typeofbox[boxstyle,right]);
 gotoxy(x7,y);write(typeofbox[boxstyle,right]);
end;

gotoxy(x1,y2);
write(typeofbox[boxstyle,loleft]);
for x:= x1+1 to x2-1 do
 write(typeofbox[boxstyle,bottom]);
write(typeofbox[boxstyle,loright]);
setattr(attrofheader);
gotoxy( (x1+x2-length(headerofwindow)) div 2,y1);
write(headerofwindow);
gotoxy(x3,y3);write(#197);
gotoxy(x4,y3);write(#197);
gotoxy(x5,y3);write(#197);
gotoxy(x6,y3);write(#197);
gotoxy(x1,y3);write(#195);
gotoxy(x2,y3);write(#180);
gotoxy(x7,y3);write(#197);
```

```
  gotoxy(x3,y1);write(#194);
  gotoxy(x4,y1);write(#194);
  gotoxy(x3,y2);write(#193);
  gotoxy(x4,y2);write(#193);
  gotoxy(x5,y1);write(#194);
  gotoxy(x6,y1);write(#194);
  gotoxy(x5,y2);write(#193);
  gotoxy(x6,y2);write(#193);
  gotoxy(x7,y1);write(#194);
  gotoxy(x7,y2);write(#193);
  setattr(attrofchar);
end;
begin
 clrscr;writeln;
 write('  no code length  no code length  no code length  no
       code length');
 write('   no.cut   st');
 windowbox(1,1,80,23,17,33,49,65,75,3);
 {gotoxy(1,1);
 write('01234567890123456789012345678901234567890123456789');
 writeln('12345678901234567890123456789');}
 gotoxy(32,1);setattr(reverselow);
 write('DIAMETER  ',copy(av[pp+1],33,9));
 setattr(lowdisplay);
 gotoxy(38,23);write(' YTP ');
end;

begin {tabl}
  window(1,1,80,25);
  clrscr;k:=0;k1:=0;l:=0;
  tef(av,pp);{made  table}window(2,4,79,22);m:=pp;
  for i:= 1 to rm do
   if rnd(ab[rb+rm+1,i],ah)>0.9 then
```

```
begin
  pc:=0;
  for j:= 1 to rm+rd do
   if ae[sa[i],j] <>0 then
    begin
      if l=0 then k:=k1
      else  k:=-1-k;
      gotoxy(wherex+16*l,wherey+k);
      if l=0 then k1:=0;{pnn(pn);}
      pc:=pc+(ae[sa[i],j]*a[j]);
      if ah[7]=1 then
       writeln(lst,'QUANTITY ':20,ae[sa[i],j]:3:0,
               ' LENGTH ':30,a[j]:2:2);
      writeln(' ',ae[sa[i],j]:3:0,'*',a[j]:2:2);
      pa:=rnd(ab[rb+rm+1,i],ah)*trunc(ae[sa[i],j]);
      ak:=a[j];pp:=m;
      jor(pn,z,pp,k,l,pa,ak,av,ah);inc(l);
      if k1<k then k1:=k;
    end;
  if ah[7]=1 then begin
   writeln(lst,'USE STEEL LENGTH ':20,pc+abs(ae[sa[i],
              rm+rd+1]):2:1,
              'QUANTITY ':30,rnd(ab[rb+rm+1,i],ah):7);
   writeln(lst,'');end;
  gotoxy(wherex+66,wherey-k-1);write(rnd(ab[rb+rm+1,i],ah):7);
  gotoxy(wherex+1,wherey);
  writeln(pc+abs(ae[sa[i],rm+rd+1]):2:1);
  t:=0;
  repeat
   inc(t);
   if ah[8+t] = (pc+abs(ae[sa[i],rm+rd+1])) then
    begin ah[12+t]:=ah[12+t]+rnd(ab[rb+rm+1,i],ah);t:=4;end
   else if ah[8+t] = 0 then
```

```
        begin ah[8+t]:= pc+abs(ae[sa[i],rm+rd+1]);
              ah[12+t]:=rnd(ab[rb+rm+1,i],ah);t:=4 end;
      until t=4;
      if (ah[7]=2) and (ah[3]=2) then ch:=readkey;
      k:=0;l:=0;
    end;
   gotoxy(wherex,wherey+k1+1);
end;
procedure ing(var a:add;z,pp,rm,t,mc:integer;var sa,b:att;
              var ab,ae:abb;var cb,ha,hb:real;var ah:hh);
var i,j,k,l,p,q,s,r,rc,rr:integer;
    ya,yb:array[1..100] of integer;
    c,ca,hd:real;

procedure inga(var z,pp,t:integer;var av:nm;var ab:abb;var ah:hh);
var i,j,n,l,k1,m,mt,b :integer;
    mc,ak:real;
    ch:char;
begin {inga}
  for i:=1 to t do
   for j:=1 to 4 do
    for n:=1 to 4 do
     if ab[i,2*j]<ab[i,2*n] then
      begin
        mc:=ab[i,2*j];ab[i,2*j]:=ab[i,2*n];ab[i,2*n]:=mc;
        mc:=ab[i,2*j-1];ab[i,2*j-1]:=ab[i,2*n-1];ab[i,2*n-1]:=mc;
      end;
  {for i:=1 to t do
   for j:=1 to 9 do write(i:2,'ab ',ab[i,j]:2:2,' ');readln;}
  for i:= 1 to t do {sum for repettition value of ab[i,t]}
   for j:= i+1 to t do
    begin
      if ab[i,9]*ab[j,9]<>0 then
```

```
      for n:= 1 to 8 do
       if ab[i,n]=ab[j,n] then inc(mt);
     if mt=8 then
      begin
        ab[i,9]:=ab[i,9]+1;ab[j,9]:=0;
      end;
     mt:=0;
   end;
k:=0;k1:=0;l:=0;
m:=pp;
for i:= 1 to t do
 if ab[i,9]>=1 then
  begin
    for j:= 1 to 4 do
     if ab[i,2*j-1] <>0 then
      begin
        if l=0 then k:=k1
        else  k:=-1-k;
        gotoxy(wherex+16*l,wherey+k);
        if l=0 then k1:=0;{pnn(pn);}
        if ah[7]=1 then
         writeln(lst,'QUNTITY ':20,ab[i,2*j-1]:2:0,
                 ' LENGTH ':30,ab[i,2*j]:4:2);
        writeln(' ',ab[i,2*j-1]:2:0,'*',ab[i,2*j]:4:2);

        pa:=ab[i,2*j-1]*ab[i,9];
        ak:=ab[i,2*j];pp:=m;
        jor(pn,z,pp,k,l,pa,ak,av,ah);inc(l);
        if k1<k then k1:=k;
      end;
    if ah[7]=1 then begin
     writeln(lst,'USE STEEL LENGTH ':20,ab[i,10]:2:1,
             'QUANTITY ':30,ab[i,9]:7:0);
```

```
unit win;

interface
uses crt,screen;
const  single      = 1;
       double      = 2;
       mix1        = 3;
       mix2        = 4;
       boxstyle          : byte     = single;
       attrofbox         : byte     = highdisplay;
       attrofwindow      : byte     = highdisplay;
       attrofheader      : byte     = highdisplay;
       attrofchar        : byte     = highdisplay;
       headerofwindow    : string   = '';
       typeofbox         : array [1..4,1..8] of char =
                     ((#196,#179,#179,#196,#218,#191,#192,#217),
                     (#205,#186,#186,#205,#201,#187,#200,#188),
                     (#205,#179,#179,#205,#213,#184,#212,#190),
                     (#196,#186,#186,#196,#214,#183,#211,#189));
var errorwindow : byte;

procedure windowbox(x1,y1,x2,y2:byte);
procedure windowopen(x1,y1,x2,y2:byte);
procedure windowclose;
procedure setboxstyle(attrib :byte);
procedure setboxattr(attrib :byte);
procedure setwinattr(attrib :byte);
procedure setheadattr(attrib :byte);
procedure setcharattr(attrib :byte);
procedure setwinheader(st :string);

implementation
type  screenline = array[1..80] of integer;
```

```
      screenarray= array[1..25] of screenline;
      screenblock= array[1..2000] of integer;
      windowlink = ^windowcontrolblock;
      windowcontrolblock = record
                             x1,y1,x2,y2 : integer;
                             x,y         : integer;
                             id          : byte;
                             backlink       : windowlink;
                             screencontents : screenblock;
                           end;
var activewindow  :windowlink;
    screenptr     :^screenarray;
    fixedsize     :integer;
    windowcount   :byte;

procedure windowbox(x1,y1,x2,y2:byte);
const top     = 1;left    = 2;
      right   = 3;bottom = 4;
      upleft  = 5;upright = 6;
      loleft  = 7;loright = 8;
var   x,y :byte;
begin
  window(x1,y1,x2,y2);
  setattr(attrofwindow);
  clrscr;
  window(1,1,80,25);
  setattr(attrofbox);
  gotoxy(x1,y1);
  write(typeofbox[boxstyle,upleft]);
  for x := x1+1 to x2-1 do
   write(typeofbox[boxstyle,top]);
  write(typeofbox[boxstyle,upright]);
  for y:= y1+1 to y2-1 do begin
```

```
    gotoxy(x1,y);write(typeofbox[boxstyle,left]);
    gotoxy(x2,y);write(typeofbox[boxstyle,right]);
  end;
  gotoxy(x1,y2);
  write(typeofbox[boxstyle,loleft]);
  for x:= x1+1 to x2-1 do
   write(typeofbox[boxstyle,bottom]);
  write(typeofbox[boxstyle,loright]);
  setattr(attrofheader);
  gotoxy( (x1+x2-length(headerofwindow)) div 2,y1);
  write(headerofwindow);
  window(x1+1,y1+1,x2-1,y2-1);
  setattr(attrofchar);
end;
procedure windowopen(x1,y1,x2,y2:byte);
var block    :windowlink;
    linelength,windowsize,i :integer;
    y:byte;
begin
  linelength :=x2-x1+1;
  windowsize:=linelength*(y2-y1+1)*2+fixedsize;
  if (x2>80) or (y2>25) or (x2-x1<2) or (y2-y1<2) then
   errorwindow:=1
  else
   if (abs(memavail) <windowsize) then
    errorwindow:=2
   else
    errorwindow:=0;
  if errorwindow =0 then
   begin
     getmem(block,windowsize);
     block^.x1 := x1;
     block^.x2 := x2;
```

```
      block^.y1 := y1;
      block^.y2 := y2;
      block^.x  := wherex;
      block^.y  := wherey;
      block^.backlink := activewindow;
      activewindow    := block;
      windowcount     := windowcount+1;
      block^.id       := windowcount;
      i:=1;
      for y:=y1 to y2 do begin
       move(screenptr^[y,x1],block^.screencontents[i],linelength*2);
       i:= i+linelength;
      end;
     windowbox(x1,y1,x2,y2);
  end;
end;

procedure windowclose;
var block :windowlink;
    linelength,windowsize,i :integer;
    y:byte;
begin
  if activewindow<>nil then
   begin
     block     := activewindow;
     linelength:= block^.x2-block^.x1+1;
     windowsize:=linelength*(block^.y2-block^.y1+1)*2+fixedsize;
     windowcount:=windowcount-1;
     i:=1;
     for y:= block^.y1 to block^.y2 do begin
       move(block^.screencontents[i],
            screenptr^[y,block^.x1],linelength*2);
       i:=i+linelength;
```

```
      end;
      activewindow := block^.backlink;
      if activewindow = nil then
       window(1,1,80,25)
      else
       with activewindow^ do window(x1+1,y1+1,x2-1,y2-1);
      gotoxy(block^.x,block^.y);
      freemem(block,windowsize);
    end;
end;

procedure initwin;
begin
  activewindow := nil;
  fixedsize    := sizeof(windowcontrolblock)-sizeof(screenblock);
  screenptr    := ptr(videoseg,0);
  window(1,1,80,25);
  windowcount:=0;
end;
procedure setboxstyle(attrib:byte);
begin
  boxstyle:=attrib;
end;
procedure setwinheader(st:string);
begin
  headerofwindow :=st;
end;
procedure setwinattr(attrib:byte);
begin
  attrofwindow:=attrib;
end;
procedure setboxattr(attrib:byte);
begin
```

```
  attrofbox:=attrib;
end;
procedure setheadattr(attrib:byte);
begin
  attrofheader:=attrib;
end;
procedure setcharattr(attrib:byte);
begin
  attrofchar:=attrib;
end;

begin
  initwin;
end.
```

```
        writeln(lst,'');end;
       gotoxy(wherex+66,wherey-k-1);write(ab[i,9]:7:0);
       gotoxy(wherex+1,wherey);writeln(ab[i,10]:2:1); b:=0;
       repeat
        inc(b);
        if ah[8+b] = ab[i,10] then
         begin ah[12+b]:=ah[12+b]+ab[i,9];b:=4; end
        else if ah[8+b] = 0 then
         begin ah[8+b]:= ab[i,10];
               ah[12+b]:=ab[i,9];b:=4 end;
       until b=4;
       if (ah[7]=2) and (ah[3]=2) then ch:=readkey;
       k:=0;l:=0;
     end;
   gotoxy(wherex,wherey+k1+1);
end;
begin{ing}
  ca:=0;cb:=0;hd:=ha;pp:=mc;
  for i:= 1 to rm+8 do
   ya[i]:=0;
  for i:= 1 to rm do
   for j:= 1 to rm do
    begin
     ya[j]:=ya[j]+trunc(ae[sa[i],j])*rnd(ab[rb+rm+1,i],ah);
     {if j=1 then
      writeln(i:2,'j ',j:2,': ',ya[j]:2,' tr',trunc(ae[sa[i],j]),
              'rb ',rb,'rnd',rnd(ab[rb+rm+1,i],ah),' ab ',
              ab[rb+rm+1,i]:2:2,'h8 ',ah[8]:2:2);}
    end;{readln;}
 { for i:=1 to rm do
   writeln(i:2,' y ',ya[i]:2,' b ',b[i]:2,' a ',a[i]:2:2,
           ' ae ',ae[sa[2],i]:2:2);}
  for i:=1 to rm do
```

```
  if ya[i]<=b[i] then
   ya[i]:=b[i]-ya[i];
  for i:=1 to rm+4 do
   yb[i]:=ya[i];rr:=rm;
  repeat
   if ya[rr]=0 then rr:=rr-1;
  until ya[rr]<>0;
 if rr<5 then
  begin
    rc:=4;
    for i:= rr+1 to 4 do
     ya[i]:=0;
   end
 else
  rc:=rr;
 {for i:=1 to rc do
  writeln(i:2,' y ',ya[i]:2,' yb ',yb[i]:2,' a ',a[i]:2:2,
          ' rc ',rc,'rr ',rr);
   readln;}
Repeat
 for i:=1 to rc do
  for p:= 0 to ya[i] do
   for j:=i+1 to rc do
    for q:= 0 to ya[j] do
     for k:=j+1 to rc do
      for r:= 0 to ya[k] do
       for l:=k+1 to rc do
        for s:= 0 to ya[l] do
   begin
     c:=p*a[i]+q*a[j]+r*a[k]+s*a[l];
     if c>0 then
     if (c>hd-1)and(c<=hd)and(yb[i]-p>=0)and(yb[j]-q>=0)and
        (yb[k]-r>=0)and(yb[l]-s>=0) then
```

```
        begin
          yb[i]:=yb[i]-p;yb[j]:=yb[j]-q;yb[k]:=yb[k]-r;
          yb[l]:=yb[l]-s;
          if c<hb then ca:=hb-c
          else         ca:=ha-c;
{ writeln('i ',i,' p ',p,' j ',j,' q ',q,' k ',k,' r ',r,' l ',l,
          ' s ',s,'c',c:2:2);}
          cb:=cb+ca;inc(t);
          ab[t,1]:=p;ab[t,2]:=a[i];ab[t,3]:=q;ab[t,4]:=a[j];
          ab[t,5]:=r;ab[t,6]:=a[k];ab[t,7]:=s;ab[t,8]:=a[l];
          ab[t,9]:=1;ab[t,10]:=ca+c;
        end;
     end;
     hd:=hd-1;
     until hd<=0;
    inga(z,pp,t,av,ab,ah);
  end;
begin{sol}
  ck:=0;pn:=0;
  for i:= 1 to rm do
   ck:=ck+abs(rnd(ab[rb+rm+1,i],ah)*ae[sa[i],rm+rd+1]);
  if ch='a' then
   begin
     mc:=pp;
     tabl(z,pp,rm,rb,rd,av,ab,ae,sa,a,ah);t:=0;
     ing(a,z,pp,rm,t,mc,sa,b,ab,ae,cb,ha,hb,ah);
     ck:=ck+abs(cb);
     if ah[7]=1 then
     writeln(lst,'       STEEL DIAMETER ',copy(av[pp],33,9),
             ' REST =',ck:4:2);
     writeln('REST STEEL =',ck:4:2);
     csum(ah);
     if (ck>c[rm+1]) and(chi='b') then
```

```
      begin
        for i:=1 to rm do
         sa[i]:=aa[i];
        tabl(z,pp,rm,rb,rd,av,ab,ae,sa,a,ah);
        if ah[7]=1 then
         writeln(lst,'       STEEL DIAMETER ',copy(av[pp],33,9),
                 ' REST =',c[rm+1]:4:2);
        writeln('REST STEEL =',c[rm+1]:4:2);
        csum(ah);
      end;
   end
  else
   begin
     tabl(z,pp,rm,rb,rd,av,ab,ae,sa,a,ah);
     if ah[7]=1 then
      writeln(lst,'       STEEL DIAMETER ',copy(av[pp],33,9),
              ' REST =',ck:4:2);
     writeln('REST STEEL =',ck:4:2);
     csum(ah);
   end;
end;{sol}

Begin {main ou4}
  ht:=0;i:=0;j:=0;k:=0;l:=0;t:=0;u:=0;v:=2;
  ra:=0;rm:=0;rn:=0;rm:=0;rb:=0;rd:=0;rn:=kt;rd:=0;
  {for i:=1 to rn do writeln(i:2,' ad ',ad[i]:4:3,' da ',
  da[i]);readln;}
  for i:=1 to 2 do
   if (af[i]*fa[i]<>0) then
    rd:=rd+1;
 swaf(rd,fa,af);
 Repeat
 if ha<hb then
```

```
 begin q:=ha;ha:=hb;hb:=q; end;
a[rn+rd+1]:=1000;
chi:='a';
if rn-ra>25 then rm:=25
else rm:=rn-ra;
for i:=ra+1 to ra+rm do
 begin
   w:=i-ra;
   a[w]:=ad[i];b[w]:=da[i];
   if a[w]>0 then
     aa[w]:=trunc(ha/a[w])
    else aa[w]:=0;
    if aa[w]>b[w] then
     aa[w]:=b[w];
 end;
 ra:=ra+rm;w:=0;
 for i:=1 to 51 do  {set 0 in variable}
  begin
    c[i]:=1000;
    for j:=1 to 50 do
     ab[i,j]:=0;
  end;
comb(fa,aa,rd,rb,rm,ha,hb,ab,a,c,af);
swac(rb,rm,ab,c);
if rb> 25 then {bound of comb}
 rb:=25;
for i:= 1 to rm do
 if a[i]>0 then {find max. of repettition value}
  begin
    if (a[i]>hb)or((ha-(trunc(ha/a[i])*a[i]))<=
       (hb-(trunc(hb/a[i])*a[i]))) then
     ht:=ha
    else ht:=hb;
```

```
    aa[i]:=trunc(ht/a[i]);
    if aa[i]>b[i] then
     begin
      aa[i]:=b[i];
      if (ha-(a[i]*b[i]))<(hb-(a[i]*b[i])) then
       ht:=ha
      else ht:=hb;
     end;
    ab[i+rb,rm+rd+1]:=(a[i]*aa[i])-ht;
  end;
for i:= 1 to rb do
 begin
  sa[i]:=i;
  ab[i,rm+rd+1]:=c[i];
 end;w:=1;
for i:= rb+1 to rb+rm do{set 1,0 }
 begin
  for j:= 1 to rm+rd do
   if (i = rb+w)and (j = w) then
    ab[i,j]:=aa[w]
   else
    ab[i,j]:=0;
  w:=w+1;
 end;
for i:= 1 to rb+rm do {set answer}
 for j:= 1 to rm+rd+1 do
  ae[i,j]:=ab[i,j];
for j:=1 to rm do
 ab[rb+rm+1,j]:=b[j];
rdf(fa,rb,rd,rm,ab,af,a); {transfer af fa}
sbt(af,rb,rm,rd,ab);      {conc. back ab for af}
if ah[5]=1 then begin out(rd,rm,rb,ab,af);readln;end;
   {For check initial value}
```

```
 for i:= 1 to rb do
  for j:= 1 to rm do
   begin
     if ab[rb+j,j]<>0 then
      ab[i,j]:=ab[i,j]/ab[rb+j,j];
     ab[i,rm+rd+1]:=ab[i,rm+rd+1]+ab[rb+j,rm+rd+1]*ab[i,j];
   End;y:=1;
 for i:= 1 to rb do
  for j:= 1 to rd do
   if (ab[i,rm+j]<>0) and (rd>0) then
    ab[i,rm+rd+1]:=ab[i,rm+rd+1]-7;
 for i:= 1 to rm do
  begin
    if ab[rb+i,i]<>0 then
     ab[rb+rm+1,i]:=ab[rb+rm+1,i]/ab[rb+i,i];
    if ae[rb+i,i]<>0 then
    ab[rb+rm+1,rm+rd+1]:=ab[rb+rm+1,rm+rd+1]+(ab[rb+rm+1,
    i]*ae[rb+i,rm+rd+1]);
    ab[rb+i,rm+rd+1]:=0;
    ab[rb+i,i]:=1;
    sa[i]:=rb+i;
  end;
 if ah[5]=1 then begin out(rd,rm,rb,ab,af);readln;end;
if rb<>0 then
 begin if (1-rm/rb>0.75) and (rm/rb<>0) and (rm/rb<>1) then
       ah[8]:=1-rm/rb else ah[8]:=0.75 end
{ begin if rm/rb<0.50 then ah[8]:=1-rm/rb else ah[8]:=rm/rb end}
     else ah[8]:=0.8;
Repeat
 w:=1;ct:=ab[1,rm+rd+1];
 for i:=1 to rb+rm do
  {if ab[i,rm+rd+1]=ct then
   begin
```

```
    mk:=0;
    for j := 1 to rm do
     mk:=mk+ab[i,j]-ab[w,j];
    if mk<0 then
     begin
       ct:=ab[i,rm+rd+1];w:=i;
     end;
  end
 else }if ab[i,rm+rd+1]<ct then
  begin
    ct:=ab[i,rm+rd+1];w:=i;
  end;
if ct<-0.009 then
 begin
   j:=1;bt:=10000;
   for j:= 1 to rm do
    if ab[w,j]>0 then
     if (ab[rb+rm+1,j]/ab[w,j])<bt then
      begin
        bt:=ab[rb+rm+1,j]/ab[w,j];
        v:=j;
      end;
   wv:=ab[w,v];t:=0;
  for i:=1 to rd do {limit not over quantity of rd-steel}
   if (wv>0)and(ab[w,rm+i]<>0) then
    if (ab[w,rm+i]*((ab[rm+rb+1,rm+i]/ab[w,rm+i])-
       (ab[rm+rb+1,v]/wv)))<0 then
     begin
       t:=1;
       ab[w,rm+rd+1]:=0;
     end;
   if (wv>0)and(t=0) then
    Begin
```

```
        if bt=10000 then bt:=0;
        sa[v]:=w;
        for i:= 1 to rm+rb+1 do
         ab[i,v]:=ab[i,v]/wv;
        for i:= 1 to rm+rb+1 do
         for j:= 1 to rm+rd+1 do
          if (ab[w,j]<>0) and(i<>w) and(j<>v)then
           begin
            ab[i,j]:=ab[w,j]*((ab[i,j]/ab[w,j])-ab[i,v]);
            if (ab[i,j]<=0.001) and(ab[i,j]>=-0.001) then ab[i,j]:=0;
           end;
        for j:=1 to rm+rd+1 do
         ab[w,j]:=0;
        ab[w,v]:=1;
      End;
  end;{writeln('w ',w,' v ',v,' ct ',ct:2:2,' wv ',wv:2:2,'t',t);
  for i:=1 to rm do write(i:2,' ',sa[i],' ');}
 if (y=2) and(ct=0) then {re-calculation y=1 ct=0}
  begin
    ch:='b';
    for i:= 1 to rm do
     begin
       str(ab[rb+rm+1,i]:6:2,p);delete(p,1,length(p)-2);
       if p <> '00' then
        ch:='a';
     end;
    if ch='a' then
     for i:= 1 to rb do
      ab[i,rm+rd+1]:=ab[i,rm+rd+1]-1
  end;inc(y);
 {out(rd,rm,rb,ab,af);write(' wv ',w,' ',v,'ct',ct:2:2);
  readln;clrscr;}
Until (ct>=0) or (y=15);
```

```
 intg(ab,c,aa,ch,chi);
 if ah[5]=1 then
  begin for i:= 1 to rm do
         write('   ',sa[i]:2);
        writeln;out(rd,rm,rb,ab,af);readln;end;
        {For check solution}
 sol(av,a,ch,chi,z,pp,kt,rb,rm,rd,b,aa,ab,ae,ah);
 Until ra>=rn;
{ for i:= 1 to rm do writeln(i:2,' sa ',sa[i]:2);}
end;

procedure data(var ah:hh;var av:nm);
type tmt=string[100];
     ti=array[1..100] of string[12];
     td=array[1..100] of byte;
     bbb=string[20];
var  key          : char;
     quit:boolean;
     aw:ti;
     st:string;
     path,et,s:bbb;
     dirin:searchrec;
     i,ptr:byte;
     ac:td;
     m,k:integer;
procedure dir(mask:bbb;var aw:ti;var i:byte);
var ef,j:string[2];
begin
  setboxstyle(mix2);
  windowopen(3,2,77,24);
  findfirst(mask,anyfile,dirin);i:=1;ef:=' ';
  write(ef);setattr(reverselow);
  write(1,' ',dirin.name);aw[i]:=dirin.name;
```

```
  setattr(lowdisplay);
  et:='                    ';delete(et,1,length(dirin.name)+3);
  write(et);
  while doserror=0 do begin
   findnext(dirin);
   if (doserror =0) then
    begin
      inc(i);str(i,j);aw[i]:=dirin.name;
      if length(j)=2 then ef:='';
      if (i mod 4=1) and (i>9) then write(' ');
      write(ef,i,' ',dirin.name);
      et:='                    ';
      if (i mod 4=0)and(i<>64) then
       writeln
      else
       begin
         str(i,s);if i= 9 then write(' ');
         delete(et,1,length(concat(s,'  ',dirin.name)));write(et);
         if i>9 then write(' ');
       end;
      aw[i]:=dirin.name;
      if i=64 then readln;
    end;
 end;
enD;
procedure mapkey(var key:char);
begin
  if funckey then
   case key of
     f1]key   : key := ^p;
     f2]key   : key := ^u;
     home]key : key := ^a;
     end]key  : key := ^z;
```

```
      lt]key    : key := ^s;
      rt]key    : key := ^d;
      dn]key    : key := ^t;
      up]key    : key := ^q
    else
      key := #00;
  end;
end;
procedure sej(var ptr:byte;var m,k:integer;var aw:ti;var ac:td);
var len:string;
    i:integer;
begin
  for i:=1 to m do
   if ac[i]=ptr then k:=1;
  if k=1 then setattr(reverselow);
  write(ptr,' ',aw[ptr]);str(ptr,len);
  gotoxy(wherex-length(concat(len,' ',aw[ptr])),wherey);
  if k=1 then setattr(lowdisplay);k:=1
end;
procedure f1key(var ptr:byte;var m:integer;var ac:td);
var i,j:integer;
begin
  j:=0;
  for i:=1 to m do
   if ac[i]=ptr then
    j:=1;
  if j=0 then
   begin
     m:=m+1;ac[m]:=ptr;
   end;
end;
procedure f2key(var ptr:byte;var m:integer;var ac:td);
var i,j:integer;
```

```
begin
  j:=0;
  for i:=1 to m do
   if (ac[i]=ptr) and(j=0) then j:=1
   else if j=1 then ac[i-1]:=ac[i];
  if j=1 then m:=m-1;
end;
procedure leftarrow(var ptr:byte;var k,m:integer;var aw:ti);
begin
  if ptr>1 then
   begin
    if (ptr mod 4 = 1) and (ptr >1) then
     begin
       sej(ptr,m,k,aw,ac);ptr:=ptr-1;
       gotoxy(wherex+3*18,wherey-1);sej(ptr,m,k,aw,ac);
     end
    else
     begin
       sej(ptr,m,k,aw,ac);ptr:=ptr-1;
       gotoxy(wherex-18,wherey);sej(ptr,m,k,aw,ac);
     end;
   end
  else begin
         sej(ptr,m,k,aw,ac);ptr:=i;
         gotoxy(18*trunc(4*frac((i+3)/4))+2,1+trunc(i/4-0.01));
         {gotoxy(18*trunc(4*frac(i/4)-1)+2,1+trunc(i/4+0.01));}
         sej(ptr,m,k,aw,ac);
       end;
end;

procedure rightarrow(var st:string; var i,ptr:byte;var k,m:integer;
                     var aw:ti);
var len:byte;
```

```
begin
  if ptr<i then
   begin
     if ptr mod 4 = 0 then
      begin
        sej(ptr,m,k,aw,ac);ptr:=ptr+1;
        gotoxy(wherex-3*18,wherey+1);sej(ptr,m,k,aw,ac);
      end
     else
      begin
        sej(ptr,m,k,aw,ac);ptr:=ptr+1;
        gotoxy(wherex+18,wherey);sej(ptr,m,k,aw,ac);
      end;
   end
  else begin
         sej(ptr,m,k,aw,ac);ptr:=1;
         gotoxy(2,1);sej(ptr,m,k,aw,ac);
       end;
end;
procedure down(var st:string; var ptr,i:byte;var k,m:integer;
               var aw:ti);
begin
  if ptr+4<=i then
  begin
    sej(ptr,m,k,aw,ac);ptr:=ptr+4;
    gotoxy(wherex,wherey+1);sej(ptr,m,k,aw,ac);
  end;
end;
procedure up(var st:string; var ptr:byte;var k,m:integer;var aw:ti);
begin
  if ptr-4>0 then
  begin
    sej(ptr,m,k,aw,ac);ptr:=ptr-4;
```

```
    gotoxy(wherex,wherey-1);sej(ptr,m,k,aw,ac);
  end;
end;
procedure processfunckey(key:char;var st:string;var i,ptr:byte;
                         var quit:boolean;
                         var m:integer;var ac:td);
var k:integer;
begin
  k:=0;
  case key of
    ^m : begin if st= '' then st:=#13; quit:=true; end;
    ^[ : begin st:= #27;quit:=true; end;
    ^s : leftarrow(ptr,k,m,aw);
    ^d : rightarrow(st,i,ptr,k,m,aw);
    ^t : down(st,ptr,i,k,m,aw);
    ^q : up(st,ptr,k,m,aw);
    ^p : f1key(ptr,m,ac);
    ^u : if m>0 then begin f2key(ptr,m,ac);k:=1 end;
  end;
end;
procedure ttt(var ha,hb:real;var af:aff;var fa:aft);
var key:char;
    quit:boolean;
procedure tt(var ha,hb:real;var af:aff;var fa:aft);
begin
  setattr(reverselow);
  gotoxy(1,25);write('F1-1SL.',ha:2:2);
  gotoxy(14,25);write('F2-2SL.',hb:2:2);
  gotoxy(27,25);write('F3-1RL.',af[1]:2:2,' Q.',fa[1]);
  gotoxy(47,25);write('F4-2RL.',af[2]:2:2,' Q.',fa[2]);
  gotoxy(68,25);write('ENTER-READY');
  setattr(lowdisplay);
end;
```

```
begin
  tt(ha,hb,af,fa);quit:=false;
  while not(quit) do
   begin
    readfunckey(key,funckey);
    case key of
     #59 : begin windowopen(25,10,55,13);setboxattr(double);
                 write('PUT 1st STEEL LENGTH : ');rer(ha);
                 windowclose;window(1,1,80,25);tt(ha,hb,af,fa);end;
     #60 : begin windowopen(25,10,55,13);setboxattr(double);
                 write('PUT 2nd STEEL LENGTH : ');rer(hb);
                 windowclose;window(1,1,80,25);tt(ha,hb,af,fa);end;
     #61 : begin windowopen(25,10,62,14);setboxattr(double);
                 write('PUT 1st REMAIN STEEL LENGTH : ');rer(af[1]);
                 write('                      QUALITY : ');rei(fa[1]);
                 windowclose;window(1,1,80,25);tt(ha,hb,af,fa);end;
     #62 : begin windowopen(25,10,60,14);setboxattr(double);
                 write('PUT 2nd REMAIN STEEL LENGTH : ');rer(af[2]);
                 write('                      QUALITY : ');rei(fa[2]);
                 windowclose;window(1,1,80,25);tt(ha,hb,af,fa);end;
     #13 : quit:=true;
   end;
  end;
end;
procedure tran(var av:nm;var aw:ti;var ac:td;ha,hb:real;
               var m:integer;var ah:hh);
var i,j,k,l,p,t,res,pt,c:integer;
    ct:real;
    fi:text;
    mk,am:string[40];
begin
  c:=0;
  for i:= 1 to m do
```

```
begin
  assign(fi,aw[ac[i]]);reset(fi);clrscr;
  while not eof(fi) and (c<100) do
   begin
     c:=c+1;
     readln(fi,av[c]);
   end;
  close(fi);
end;
if c>850 then c:=850;  {set limited}
 for i:= 1 to c do  {separate type of steel}
  for j:= i+1 to c do
   begin
     if copy(av[i],33,9)=copy(av[j],33,9) then
      begin
        mk:=av[i+1];av[i+1]:=av[j];av[j]:=mk;
      end;
   end;
am:=av[1];av[c+1]:='';k:=1;
for i:=2 to c+1 do
 if copy(am,33,9)<>copy(av[i],33,9) then
  begin
   for j:=k to i-1 do
    for l:=j+1 to i-1 do
     begin
       val(copy(av[j],20,4),p,res);val(copy(av[l],20,4),t,res);
       if p>t then
        begin
          mk:=av[l];av[l]:=av[j];av[j]:=mk;
        end;
     end;
     k:=i;am:=av[i];
  end;
```

```
    for i:=1 to c do
     writeln(i:3,' ',av[i]);
    for i:=1 to c do
     da[i]:=0;
    val(copy(av[1],20,4),ct,res);ct:=ct/100;
    ad[1]:=ct;kt:=0;
    am:=av[1];av[c+1]:='';pp:=0;
    for i:=1 to c+1 do
     begin
       if copy(am,33,9)<>copy(av[i],33,9) then
        begin
          z:=i;window(1,1,80,25);
          if ah[3]<>0 then ttt(ha,hb,af,fa);
          gotoxy(34,12);setattr(reverselow);write('WAIT');
          setattr(lowdisplay);
          ou4(av,da,ad,z,pp,kt,ha,hb,ah,af,fa);am:=av[i];kt:=0;
          for t:=1 to 2 do
           begin af[t]:=0;fa[t]:=0; end;
          if (i=c+1) and (ah[3]<>0) then readln;
        end;
       if (copy(am,20,4)<>copy(av[i],20,4)) or (kt=0) then
        begin
          val(copy(av[i],20,4),ct,res);ct:=ct/100;
          inc(kt);ad[kt]:=ct;da[kt]:=0;am:=av[i];
        end;
       val(copy(av[i],25,6),pt,res);
       da[kt]:=da[kt]+pt;
     end;
end;

begin
  ha:=ah[1];hb:=ah[2];af[1]:=0;fa[1]:=0;af[2]:=0;fa[2]:=0;
  windowopen(1,0,80,25);
```

```
  setattr(reverselow);
  gotoxy(3,1);write('ENTER OR ESC-FINISH');
  gotoxy(26,1);write('F1-CHOSE FILE');
  gotoxy(44,1);write('F2-REJECT FILE');
  gotoxy(64,1);write(#18,'-MOVE CURSOR');
  setattr(lowdisplay);
  path:='*.txt';m:=0;
  dir(path,aw,i);
  gotoxy(2,1);quit:=false;
  st:=key;ptr:=1;
  while not(quit) do
   begin
     readfunckey(key,funckey);
     mapkey(key);
     processfunckey(key,st,i,ptr,quit,m,ac);
   end;
  tran(av,aw,ac,ha,hb,m,ah);
  windowclose;
end;
begin
end.

{for i:=1 to rn-1 do
   for j:=i+1 to rn do
    if abs(ad[i]-ad[j])<0.01 then
     begin
       da[i]:=da[i]+da[j];
       for k:=j to rn-1 do
        begin
          ad[k]:=ad[k+1];da[k]:=da[k+1];
        end;
       rn:=rn-1;
     end;}
```

```
unit screen;
interface
uses crt,dos;
const nodisplay        = $00;
      lowdisplay       = $07;
      highdisplay      = $0F;
      underlinelow     = $01;
      underlinehigh    = $09;
      reverselow       = $70;
      reversehigh      = $78;
      blinklow         = $87;
      blinkhigh        = $8F;
      unblinklow       = $81;
      unblinkhigh      = $89;
      revblinklow      = $F0;
      revblinkhigh     = $F8;
      colorseg         = $B800;
      monoseg          = $B000;
type  agt = array[1..850] of integer;
     att = array[1..60] of integer;
var   da:agt;
      b,sa,aa:att;
      videoseg         : word;
      crttype           : byte absolute $0040:$0049;
      cursormode       : word absolute $0040:$0060;
      vport            : word absolute $0040:$0063;

procedure setattr(attrib:byte);
procedure setcursor(top,bottom:byte);
procedure cursoron;
procedure cursoroff;

implementation
```

```
 var regs :registers;
 procedure setattr(attrib:byte);
 begin
   textattr:=attrib;
 end;
 procedure setcursor(top,bottom:byte);
 begin
   regs.ah := 1;
   regs.ch := top;
   regs.cl := bottom;
   intr($10,regs);
end;
procedure cursoron;
begin
  port[vport]    := 10;
  port[vport+1]  := Hi(cursormode) and $DF;
  port[vport]    := 11;
  port[vport+1]  := lo(cursormode)
end;
procedure cursoroff;
begin
  port[vport]    := 10;
  port[vport+1]  := Hi(cursormode) or $20;
  port[vport]    := 11;
  port[vport+1]  := lo(cursormode)
end;
procedure identifycrt;
begin
  case crttype of
    0..3  : videoseg := colorseg;
       7  : videoseg := monoseg;
 end;
end;
```

```
begin
  identifycrt;
end.
```

```
unit keyboard;

interface
uses crt,vav;
var key  : char;
    funckey:boolean;
    ae:abb;
const  return]key = #13; esc]key = #27;
       shift]tab  = #15;
       alt]q]key  = #16; alt]w]key  = #17;
       alt]e]key  = #18; alt]r]key  = #19;
       alt]t]key  = #20; alt]y]key  = #21;
       alt]u]key  = #22; alt]i]key  = #23;
       alt]o]key  = #24; alt]p]key  = #25;

       alt]a]key  = #30; alt]s]key  = #31;
       alt]d]key  = #32; alt]f]key  = #33;
       alt]g]key  = #34; alt]h]key  = #35;
       alt]j]key  = #36; alt]k]key  = #37;
       alt]l]key  = #38;
       alt]z]key  = #44; alt]x]key  = #45;
       alt]c]key  = #46; alt]v]key  = #47;
       alt]b]key  = #48; alt]n]key  = #49;
       alt]m]key  = #50;

       f1]key  = #59; f2]key  = #60;
       f3]key  = #61; f4]key  = #62;
       f5]key  = #63; f6]key  = #64;
       f7]key  = #65; f8]key  = #66;
       f9]key  = #67; f10]key = #68;

       home]key  = #71; up]key   = #72;
       pgup]key  = #73; lt]key   = #75;
```

```
rt]key     = #77; end]key  = #79;
dn]key     = #80; pgdn]key = #81;
ins]key    = #82; del]key  = #83;

shift]f1]key  = #84; shift]f2]key  = #85;
shift]f3]key  = #86; shift]f4]key  = #87;
shift]f5]key  = #88; shift]f6]key  = #89;
shift]f7]key  = #90; shift]f8]key  = #91;
shift]f9]key  = #92; shift]f10]key = #93;

ctrl]f1]key  = #94;  ctrl]f2]key  = #95;
ctrl]f3]key  = #96;  ctrl]f4]key  = #97;
ctrl]f5]key  = #98;  ctrl]f6]key  = #99;
ctrl]f7]key  = #100; ctrl]f8]key  = #101;
ctrl]f9]key  = #102; ctrl]f10]key = #103;

alt]f1]key  = #104; alt]f2]key  = #105;
alt]f3]key  = #106; alt]f4]key  = #107;
alt]f5]key  = #108; alt]f6]key  = #109;
alt]f7]key  = #110; alt]f8]key  = #111;
alt]f9]key  = #112; alt]f10]key = #113;

ctrl]prtsc]key  = #114; ctrl]lt]key    = #115;
ctrl]rt]key     = #116; ctrl]end]key   = #117;
ctrl]pgdn]key   = #118; ctrl]home]key  = #119;

alt]1]key  = #120; alt]2]key  = #121;
alt]3]key  = #122; alt]4]key  = #123;
alt]5]key  = #124; alt]6]key  = #125;
alt]7]key  = #126; alt]8]key  = #127;
alt]9]key  = #128; alt]10]key = #129;

ctrl]pgup]key   = #132;
```

```
        f11]key          = #133; f12]key          = #134;
        shift]f11]key    = #135; shift]f12]key    = #136;
        ctrl]f11]key     = #137; ctrl]f12]key     = #138;
        alt]f11]key      = #139; alt]12]key       = #140;

procedure readfunckey(var key :char;var funckey:boolean);
procedure readstring(var st:string);
procedure readintnum(var num,code : integer);
procedure readrealnum(var num:real;var code:integer);

implementation
procedure readfunckey(var key:char;var funckey:boolean);
begin
  key:=readkey;
  if key= #0 then
   begin
     funckey :=true;
     key     :=readkey;
   end
  else
   begin
   funckey := false;
   end;
end;
procedure mapkey(var key:char);
begin
  if funckey then
   case key of
     home]key : key := ^a;
     end]key  : key := ^z;
     lt]key   : key := ^s;
     rt]key   : key := ^d;
     del]key  : key := ^g
```

```
    else
      key := #00;
  end;
end;

procedure backspace(var st:string; var ptr:byte);
var tail:string;
begin
  if ptr <> 1 then
   begin
     tail:=copy(st,ptr,length(st)-ptr+1);
     delete(st,ptr-1,1);
     ptr:=ptr-1;
     write(^h);
     clreol;
     write(tail);
     gotoxy(wherex-length(tail),wherey);
   end;
end;
procedure leftarrow(var ptr:byte);
begin
  if ptr<>1 then
   begin
     ptr:=ptr-1;
     write(^h);
  end;
end;

procedure rightarrow(st:string; var ptr:byte);
var len:byte;
begin
  len:=length(st);
  if (ptr<=len) and(len<>0) then
```

```
  begin
    ptr:=ptr+1;
    gotoxy(wherex+1,wherey);
  end;
end;
procedure tohome(var ptr:byte);
begin
  gotoxy(wherex-ptr+1,wherey);
  ptr:=1;
end;
procedure toend(st:string; var ptr:byte);
var len:byte;
begin
  len:=length(st);
  if ptr<=len then
  begin
    tohome(ptr);
    gotoxy(wherex+len,wherey);
    ptr:=len+1;
  end;
end;
procedure truncate(st:string; var ptr:byte);
begin
  st:=copy(st,1,ptr-1);
  ptr:=length(st)+1;
  if ptr = 1 then st:= '';
  clreol;
end;
procedure del(st:string; var ptr:byte);
var len:byte;
    tail:string;
begin
  len:=length(st);
```

```
  if (ptr<=len) and(len<>0) then
  begin
    tail:=copy(st,ptr+1,length(st)-ptr);
    delete(st,ptr,1);
    clreol;
    write(tail);
    gotoxy(wherex-length(st),wherey);
  end;
end;
procedure clear(st:string; var ptr:byte);
begin
  tohome(ptr);
  clreol;
  st:='';
end;
procedure character(var st:string; ch:char;var ptr:byte);
var p,x,len:byte;
begin
  len:=length(st);
  if ptr > len then
   begin
     if(wherex<=(lo(windmax)-lo(windmin)) )then
      begin
        st:=st+ch;
        ptr:=ptr+1;
        write(ch);
      end;
    end
   else
    if (len< (lo(windmax)-lo(windmin)) ) then
     begin
       x:=wherex;
       p:=ptr;
```

```
        insert(ch,st,ptr);
        ptr:=ptr+1;
        tohome(p);
        write(st);
        gotoxy(x+1,wherey);
      end;
end;
procedure processfunckey(key:char;var st:string;var ptr:byte;
var quit:boolean);
begin
  case key of
    ^m : begin if st= '' then st:=#13; quit:=true; end;
    ^[ : begin st:= #27;quit:=true; end;
    ^h : backspace(st,ptr);
    ^g : del(st,ptr);
    ^s : leftarrow(ptr);
    ^d : rightarrow(st,ptr);
    ^t : truncate(st,ptr);
    ^a : tohome(ptr);
    ^z : toend(st,ptr);
    ^y : clear(st,ptr);
  else
   if key >=#32 then
    character(st,key,ptr);
  end;
end;
procedure readstring(var st:string);
var ch:char;
    x,y,ptr:byte;
    quit:boolean;
begin
  quit :=false;
  ptr  :=1;
```

```
  x    :=wherex;
  y    :=wherey;
  write(st);
  gotoxy(x,y);
  readfunckey(key,funckey);
  mapkey(key);
  if (key<#32) then
   processfunckey(key,st,ptr,quit)
  else
  begin
    st  :=key;
    ptr :=2;
    gotoxy(x,y);
    clreol;
    write(st);
  end;
 while not(quit) do
 begin
   readfunckey(key,funckey);
   mapkey(key);
   processfunckey(key,st,ptr,quit);
 end;
end;
procedure readintnum(var num,code : integer);
var st:string;
begin
  str(num,st);
  readstring(st);
  val(st,num,code);
  if code<>0 then code:=255;
  if (st = #27) or (st =#13) then code:=1;
end;
procedure readrealnum(var num:real; var code:integer);
```

```
var st:string;
begin
  str(num:1:12,st);
  while st[length(st)] = '0' do st[0] :=chr(ord(st[0])-1);
  readstring(st);
  st:='0'+st;
  val(st,num,code);
  if code<>0 then code:=255;
  if (st=#27) or (st=#13) then code :=1;
end;
end.
```

```
unit win;

interface
uses crt,screen;
const  single      = 1;
       double      = 2;
       mix1        = 3;
       mix2        = 4;
       boxstyle          : byte     = single;
       attrofbox         : byte     = highdisplay;
       attrofwindow      : byte     = highdisplay;
       attrofheader      : byte     = highdisplay;
       attrofchar        : byte     = highdisplay;
       headerofwindow    : string   = '';
       typeofbox         : array [1..4,1..8] of char =
                    ((#196,#179,#179,#196,#218,#191,#192,#217),
                    (#205,#186,#186,#205,#201,#187,#200,#188),
                    (#205,#179,#179,#205,#213,#184,#212,#190),
                    (#196,#186,#186,#196,#214,#183,#211,#189));
var errorwindow : byte;

procedure windowbox(x1,y1,x2,y2:byte);
procedure windowopen(x1,y1,x2,y2:byte);
procedure windowclose;
procedure setboxstyle(attrib :byte);
procedure setboxattr(attrib :byte);
procedure setwinattr(attrib :byte);
procedure setheadattr(attrib :byte);
procedure setcharattr(attrib :byte);
procedure setwinheader(st :string);

implementation
type  screenline = array[1..80] of integer;
```

```
      screenarray= array[1..25] of screenline;
      screenblock= array[1..2000] of integer;
      windowlink = ^windowcontrolblock;
      windowcontrolblock = record
                             x1,y1,x2,y2 : integer;
                             x,y         : integer;
                             id          : byte;
                             backlink       : windowlink;
                             screencontents : screenblock;
                           end;
var activewindow  :windowlink;
    screenptr     :^screenarray;
    fixedsize     :integer;
    windowcount   :byte;

procedure windowbox(x1,y1,x2,y2:byte);
const top     = 1;left    = 2;
      right   = 3;bottom = 4;
      upleft  = 5;upright = 6;
      loleft  = 7;loright = 8;
var   x,y :byte;
begin
  window(x1,y1,x2,y2);
  setattr(attrofwindow);
  clrscr;
  window(1,1,80,25);
  setattr(attrofbox);
  gotoxy(x1,y1);
  write(typeofbox[boxstyle,upleft]);
  for x := x1+1 to x2-1 do
   write(typeofbox[boxstyle,top]);
  write(typeofbox[boxstyle,upright]);
  for y:= y1+1 to y2-1 do begin
```

```
   gotoxy(x1,y);write(typeofbox[boxstyle,left]);
   gotoxy(x2,y);write(typeofbox[boxstyle,right]);
  end;
  gotoxy(x1,y2);
  write(typeofbox[boxstyle,loleft]);
  for x:= x1+1 to x2-1 do
   write(typeofbox[boxstyle,bottom]);
  write(typeofbox[boxstyle,loright]);
  setattr(attrofheader);
  gotoxy( (x1+x2-length(headerofwindow)) div 2,y1);
  write(headerofwindow);
  window(x1+1,y1+1,x2-1,y2-1);
  setattr(attrofchar);
end;
procedure windowopen(x1,y1,x2,y2:byte);
var block    :windowlink;
    linelength,windowsize,i :integer;
    y:byte;
begin
  linelength :=x2-x1+1;
  windowsize:=linelength*(y2-y1+1)*2+fixedsize;
  if (x2>80) or (y2>25) or (x2-x1<2) or (y2-y1<2) then
   errorwindow:=1
  else
   if (abs(memavail) <windowsize) then
    errorwindow:=2
   else
    errorwindow:=0;
  if errorwindow =0 then
   begin
     getmem(block,windowsize);
     block^.x1 := x1;
     block^.x2 := x2;
```

```
      block^.y1 := y1;
      block^.y2 := y2;
      block^.x  := wherex;
      block^.y  := wherey;
      block^.backlink := activewindow;
      activewindow    := block;
      windowcount     := windowcount+1;
      block^.id       := windowcount;
      i:=1;
      for y:=y1 to y2 do begin
       move(screenptr^[y,x1],block^.screencontents[i],linelength*2);
       i:= i+linelength;
      end;
     windowbox(x1,y1,x2,y2);
  end;
end;

procedure windowclose;
var block :windowlink;
    linelength,windowsize,i :integer;
    y:byte;
begin
  if activewindow<>nil then
   begin
     block     := activewindow;
     linelength:= block^.x2-block^.x1+1;
     windowsize:=linelength*(block^.y2-block^.y1+1)*2+fixedsize;
     windowcount:=windowcount-1;
     i:=1;
     for y:= block^.y1 to block^.y2 do begin
       move(block^.screencontents[i],
            screenptr^[y,block^.x1],linelength*2);
       i:=i+linelength;
```

```
      end;
      activewindow := block^.backlink;
      if activewindow = nil then
       window(1,1,80,25)
      else
       with activewindow^ do window(x1+1,y1+1,x2-1,y2-1);
      gotoxy(block^.x,block^.y);
      freemem(block,windowsize);
    end;
end;

procedure initwin;
begin
  activewindow := nil;
  fixedsize    := sizeof(windowcontrolblock)-sizeof(screenblock);
  screenptr    := ptr(videoseg,0);
  window(1,1,80,25);
  windowcount:=0;
end;
procedure setboxstyle(attrib:byte);
begin
  boxstyle:=attrib;
end;
procedure setwinheader(st:string);
begin
  headerofwindow :=st;
end;
procedure setwinattr(attrib:byte);
begin
  attrofwindow:=attrib;
end;
procedure setboxattr(attrib:byte);
begin
```

```
  attrofbox:=attrib;
end;
procedure setheadattr(attrib:byte);
begin
  attrofheader:=attrib;
end;
procedure setcharattr(attrib:byte);
begin
  attrofchar:=attrib;
end;

begin
  initwin;
end.
```

ประวัติผู้เขียน

นายโยธิน ดรีรัตนพันธ์ เกิดวันที่ 31 กรกฎาคม พ.ศ. 2509 ที่อำเภอเมือง จังหวัดพัทลุง สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตรบัณฑิต ภาควิศวกรรมโยธา มหาวิทยาลัยสงขลานครินทร์ ในปีการศึกษา 2531 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต ที่จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2532