



รายงานการเขียนโปรแกรมคำสั่ง ARM (Project 1)

เสนอ

ผู้ช่วยศาสตราจารย์ ดร.ยุทธนา เจวจินดา

ผู้จัดทำ

นางสาววิมลสิริ อินทร์บำรุง

รหัสนักศึกษา 630910653

ชั้นปีที่ 2

รายงานเล่มนี้เป็นส่วนหนึ่งในวิชาสถาปัตยกรรมและองค์ประกอบระบบคอมพิวเตอร์

(Computer System Architecture and Organization) รหัสวิชา 618242

ภาคเรียนที่ 2 ปีการศึกษา 2564

มหาวิทยาลัยศิลปากร

อธิบายปัญหาที่ต้องการแก้

ต้องการคิดเกรดจากคะแนนรวมทั้งหมดที่ทำการรวมคะแนนในส่วนต่างๆแล้ว และแสดงผลของเกรดที่ได้ที่หน้าจอ

โดยมีเกณฑ์ของคะแนนในแต่ละส่วนต่างๆ ดังนี้

คะแนนรวม 100%	
คะแนนเข้าชั้นเรียน	5%
คะแนนการบ้าน	10%
คะแนนงานProject 1 (midterm)	20%
คะแนนงานProject 2 (final)	30%
คะแนนสอบปลายภาค	35%

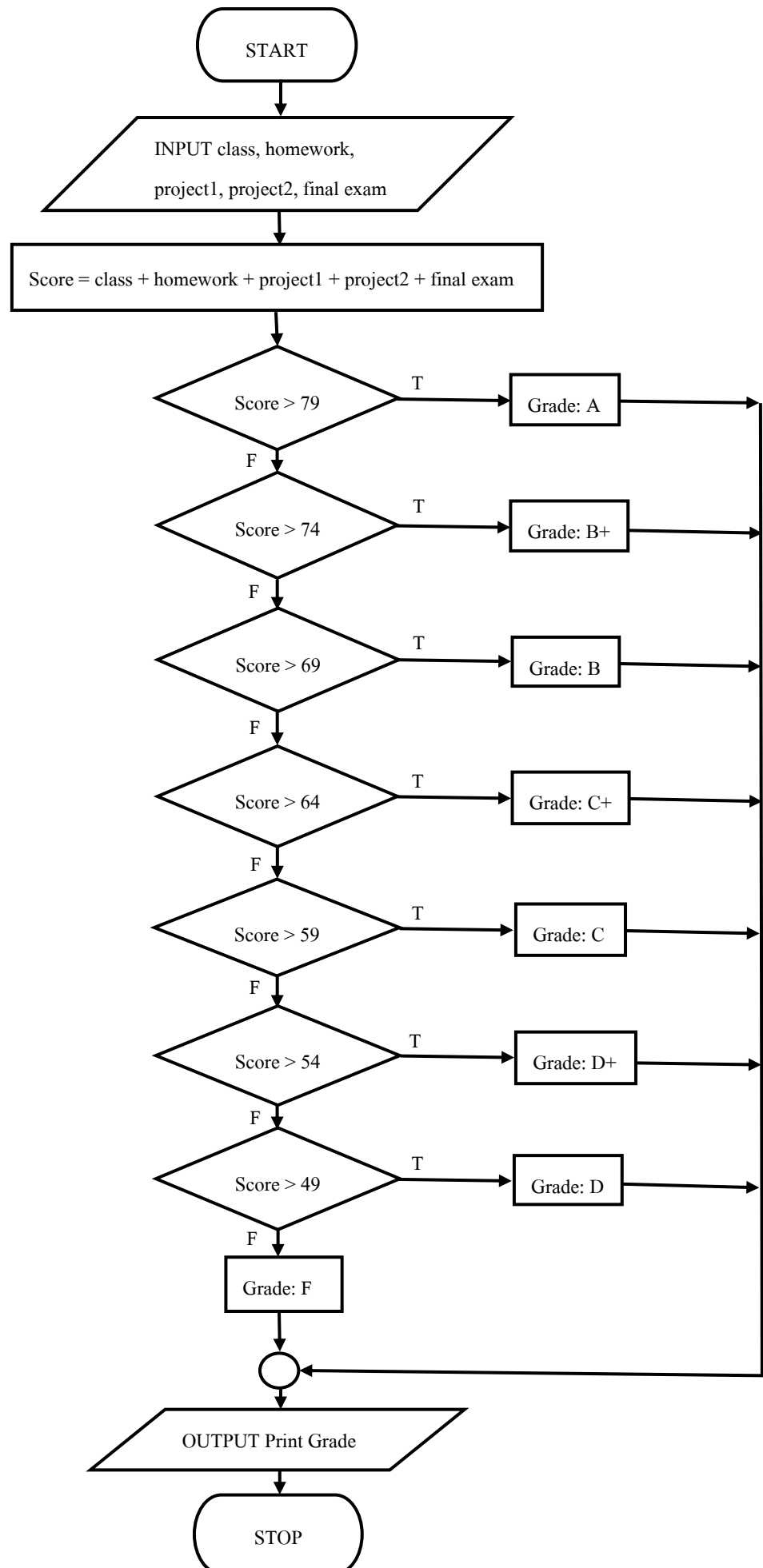
และมีเกณฑ์การคิดเกรด ดังนี้

การคิดเกรด	
A	80 – 100
B+	75 – 79
B	70 – 74
C+	65 – 69
C	60 – 64
D+	55 – 59
D	50 – 54
F	0 – 49

โดยกำหนดให้ R1 เก็บคะแนนการเข้าเรียน, R2 เก็บคะแนนการบ้าน, R3 เก็บคะแนนProject1, R4 เก็บคะแนนProject2 และR5 เก็บคะแนนสอบปลายภาค

จากนั้นทำการรวมคะแนนในส่วนต่างๆ แล้วนำค่าที่ได้ไปเก็บไว้ใน R0 ต่อมาจะทำการคิดเกรดโดยเริ่มเปรียบเทียบที่ A โดยเปรียบเทียบว่าค่าใน R0 > 79 หรือไม่ (โดยในภาษา Assembly จะใช้คำสั่งตรงข้ามกับภาษา C) ถ้ามากกว่าก็จะข้ามไปทำงานที่ gradeA เพื่อเปลี่ยนช่องว่างในส่วนของค่าในตัวแปร grade จาก “Grade: ” ให้เป็น “Grade:A ” และทำงานต่อไปที่ output เพื่อแสดงค่า “Grade:A ” ที่จอภาพ แต่ถ้า R0<=79 ก็จะทำงานต่อไปที่ Bplus และเปรียบเทียบค่าใน R0 ว่ามากกว่าเกณฑ์ของเกรดนั้นหรือไม่ โดยมีการทำงานคล้ายเดิม แต่เปลี่ยนตัวแปรตามเกณฑ์การคิดเกรด

Flowchart



โปรแกรม

```
QEMU
GNU nano 2.2.6      File: pj1.s      Modified

.global _start
_start:
MOV R1,#5
MOV R2,#10
MOV R3,#20
MOV R4,#30
MOV R5,#35
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
X Exit      ^J Justify   ^W Where Is   ^U Next Page  ^U UnCut Text ^T To Spell
```

```
QEMU
GNU nano 2.2.6      File: pj1.s      Modified

BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
BLS C
B gradeCplus
C:
CMP R0,#59
BLS Dplus
B gradeC
Dplus:
CMP R0,#54
BLS D
B gradeDplus
D:
CMP R0,#49
BLS gradeF
G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
X Exit      ^J Justify   ^W Where Is   ^U Next Page  ^U UnCut Text ^T To Spell
```

```
QEMU
GNU nano 2.2.6 File: pj1.s Modified

B gradeCplus
C:
CMP R0,#59
BLS Dplus
B gradeC
Dplus:
CMP R0,#54
BLS D
B gradeDplus
D:
CMP R0,#49
BLS gradeF
B gradeD
gradeA:
MOV R0,#0x41
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeBplus:
MOV R0,#0x42
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^U Next Page ^U UnCut Text ^T To Spell
```

```
QEMU
GNU nano 2.2.6 File: pj1.s Modified

gradeA:
MOV R0,#0x41
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeBplus:
MOV R0,#0x42
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]
B output
gradeB:
MOV R0,#0x42
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeCplus:
MOV R0,#0x43
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]
```

```
QEMU
GNU nano 2.2.6 File: pj1.s Modified

gradeB:
MOV R0,#0x42
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeCplus:
MOV R0,#0x43
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]
B output
gradeC:
MOV R0,#0x43
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeDplus:
MOV R0,#0x44
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^U Next Page ^U UnCut Text ^T To Spell
```

```
QEMU
GNU nano 2.2.6 File: pj1.s Modified

gradeC:
MOV R0,#0x43
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeDplus:
MOV R0,#0x44
LDR R1,=grade
STRB R0,[R1,#6]
MOV R0,#0x2B
LDR R1,=grade
STRB R0,[R1,#7]
B output
gradeD:
MOV R0,#0x44
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeF:
MOV R0,#0x46
LDR R1,=grade
STRB R0,[R1,#6]
output:
MOV R7,#4
MOV R0,#1

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^U Next Page ^U UnCut Text ^T To Spell
```

QEMU

GNU nano 2.2.6 File: pj1.s

```
gradeD:
MOV R0,#0x44
LDR R1,=grade
STRB R0,[R1,#6]
B output
gradeF:
MOV R0,#0x46
LDR R1,=grade
STRB R0,[R1,#6]
output:
MOV R7,#4
MOV R0,#1
MOV R2,#9
LDR R1,=grade
SWI 0
exit:
MOV R7,#1
SWI 0
.data
grade:
.ascii "Grade: \n"
```

[Wrote 99 lines]

^G Get Help	^O WriteOut	^R Read File	^Y Prev Page	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where Is	^V Next Page	^U UnCut Text	^T To Spell

อธิบายการทำงานของโปรแกรม

.global _start ; เป็นจุดเริ่มต้นการทำงาน โดยกำหนดให้เริ่มทำงานที่ _start: เป็นต้นไป

_start:

MOV R1, #5 ; นำ 5 ไปเก็บไว้ใน R1 (R1 <- 5)

MOV R2, #10 ; นำ 10 ไปเก็บไว้ใน R2 (R2 <- 10)

MOV R3, #20 ; นำ 20 ไปเก็บไว้ใน R3 (R3 <- 20)

MOV R4, #30 ; นำ 30 ไปเก็บไว้ใน R4 (R4 <- 30)

MOV R5, #35 ; นำ 35 ไปเก็บไว้ใน R5 (R5 <- 35)

ADD R6, R1, R2 ; นำค่าใน R1 บวกกับค่าใน R2 แล้วนำค่าที่ได้มาเก็บไว้ใน R6
(R6 <- R1+R2) (15 = 5 + 10)

ADD R7, R3, R4 ; นำค่าใน R3 บวกกับค่าใน R4 แล้วนำค่าที่ได้มาเก็บไว้ใน R7
(R7 <- R3+R4) (50 = 20 + 30)

ADD R8, R5, R6 ; นำค่าใน R5 บวกกับค่าใน R6 แล้วนำค่าที่ได้มาเก็บไว้ใน R8
(R8 <- R5+R6) (50 = 35 + 15)

ADD R0, R7, R8 ; นำค่าใน R7 บวกกับค่าใน R8 แล้วนำค่าที่ได้มาเก็บไว้ใน R0
(R0 <- R7+R8) (100 = 50 + 50)

A:

CMP R0, #79 ; นำค่าใน R0 มาเทียบกับ 79 (R0 <= 79?)

BLS Bplus ; ถ้าค่าใน R0 <= 79 จะทำงานต่อที่ Bplus T (R0 <= 79)

B gradeA ; ถ้าค่าใน R0 > 79 จะทำงานข้ามไปที่ gradeA F (R0 > 79)

Bplus:

CMP R0, #74 ; นำค่าใน R0 มาเทียบกับ 74 (R0 <= 74?)

BLS B ; ถ้าค่าใน R0 <= 74 จะทำงานต่อที่ B T (R0 <= 74)

B gradeBplus ; ถ้าค่าใน R0 > 74 จะทำงานข้ามไปที่ gradeBplus F (R0 > 74)

B:

CMP R0, #69 ; นำค่าใน R0 มาเทียบกับ 69 (R0 <= 69?)

BLS Cplus ; ถ้าค่าใน R0 <= 69 จะทำงานต่อที่ Cplus T (R0 <= 69)

B gradeB ; ถ้าค่าใน R0 > 69 จะทำงานข้ามไปที่ gradeB F (R0 > 69)

Cplus:

CMP R0, #64	; นำค่าในR0 มาเทียบกับ 64	(R0 <= 64?)
BLS C	; ถ้าค่าใน R0 <= 64 จะทำงานต่อที่ C	T (R0 <= 64)
B gradeCplus	; ถ้าค่าใน R0 > 64 จะทำงานข้ามไปที่ gradeCplus	F (R0 > 64)

C:

CMP R0, #59	; นำค่าในR0 มาเทียบกับ 59	(R0 <= 59?)
BLS Dplus	; ถ้าค่าใน R0 <= 59 จะทำงานต่อที่ Dplus	T (R0 <= 59)
B gradeC	; ถ้าค่าใน R0 > 59 จะทำงานข้ามไปที่ gradeC	F (R0 > 59)

Dplus:

CMP R0, #54	; นำค่าในR0 มาเทียบกับ 54	(R0 <= 54?)
BLS D	; ถ้าค่าใน R0 <= 54 จะทำงานต่อที่ D	T (R0 <= 54)
B gradeDplus	; ถ้าค่าใน R0 > 54 จะทำงานข้ามไปที่ gradeDplus	F (R0 > 54)

D:

CMP R0, #49	; นำค่าในR0 มาเทียบกับ 49	(R0 <= 49?)
BLS gradeF	; ถ้าค่าใน R0 <= 49 จะทำงานต่อที่ gradeF	T (R0 <= 49)
B gradeD	; ถ้าค่าใน R0 > 49 จะทำงานข้ามไปที่ gradeD	F (R0 > 49)

gradeA:

MOV R0, #0x41	; (R0 <- A)	} แทนค่าที่เก็บใน R0 นั้นคือ A ที่ ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade		
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]	
B output	; ข้ามการทำงานไปที่ output	

gradeBplus:

MOV R0, #0x42	; (R0 <- B)	} แทนค่าที่เก็บใน R0 นั้นคือ B ที่ ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade		
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]	
MOV R0, #0x2B	; (R0 <- +)	} แทนค่าที่เก็บใน R0 นั้นคือ + ที่ ตำแหน่งที่ 7 ในค่าตัวแปร grade
LDR R1, =grade		
STRB R0, [R1, #7]	; M [R1+7] <- R0 [7:0]	
B output	; ข้ามการทำงานไปที่ output	

gradeB:

MOV R0, #0x42	; (R0 <- B)	}	แทนค่าที่เก็บใน R0 นั้นคือ B ที่ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]		
B output	; ข้ามการทำงานไปที่ output		
gradeCplus:			
MOV R0, #0x43	; (R0 <- C)	}	แทนค่าที่เก็บใน R0 นั้นคือ C ที่ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]		
MOV R0, #0x2B	; (R0 <- +)	}	แทนค่าที่เก็บใน R0 นั้นคือ + ที่ตำแหน่งที่ 7 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #7]	; M [R1+7] <- R0 [7:0]		
B output	; ข้ามการทำงานไปที่ output		
gradeC:			
MOV R0, #0x43	; (R0 <- C)	}	แทนค่าที่เก็บใน R0 นั้นคือ C ที่ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]		
B output	; ข้ามการทำงานไปที่ output		
gradeDplus:			
MOV R0, #0x44	; (R0 <- D)	}	แทนค่าที่เก็บใน R0 นั้นคือ D ที่ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]		
MOV R0, #0x2B	; (R0 <- +)	}	แทนค่าที่เก็บใน R0 นั้นคือ + ที่ตำแหน่งที่ 7 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #7]	; M [R1+7] <- R0 [7:0]		
B output	; ข้ามการทำงานไปที่ output		
gradeD:			
MOV R0, #0x44	; (R0 <- D)	}	แทนค่าที่เก็บใน R0 นั้นคือ D ที่ตำแหน่งที่ 6 ในค่าตัวแปร grade
LDR R1, =grade			
STRB R0, [R1, #6]	; M [R1+6] <- R0 [7:0]		

```

B output          ; ข้ามการทำงานไปที่ output
gradeF:
    MOV R0, #0x46      ; (R0 <- F)
    LDR R1, =grade
    STRB R0, [R1, #6]   ; M [R1+6] <- R0 [7:0]
}   แทนค่าที่เก็บใน R0 นั้นคือ F ที่
    ตำแหน่งที่ 6 ในค่าตัวแปร grade

output:
    MOV R7, #4
    MOV R0, #1
    MOV R2, #9
    LDR R1, =grade
    SWI 0
}   ส่วนที่นำค่าที่อยู่ในตัวแปร grade มาแสดงผลที่จอภาพ ตั้งแต่บิต 0 ถึง 9

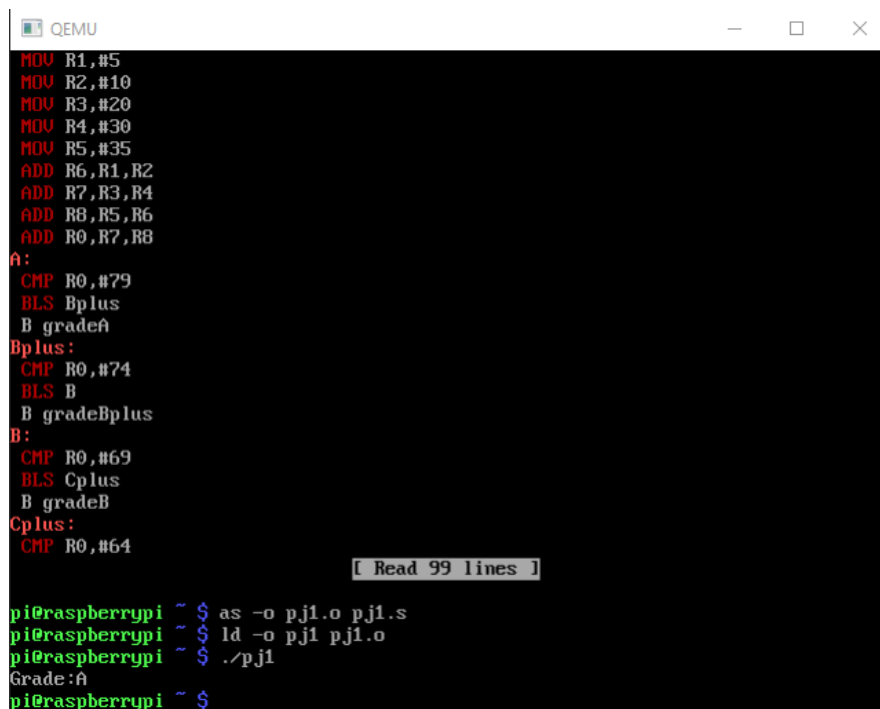
exit:
    MOV R7, #1
    SWI 0
}   ส่วนที่นำค่าที่อยู่ใน R0 มาแสดงผลที่จอภาพ ตั้งแต่บิต 0 ถึง 7 ( R0 [7:0] )

.data          ; ส่วนของข้อมูล

grade:
    .ascii "Grade: \n"
}   สร้างตัวแปร grade เก็บ "Grade: \n"

```

ผลการทำงาน



```
MOV R1,#5
MOV R2,#10
MOV R3,#20
MOV R4,#30
MOV R5,#35
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64

[ Read 99 lines ]

pi@raspberrypi ~ $ as -o pj1.o pj1.s
pi@raspberrypi ~ $ ld -o pj1 pj1.o
pi@raspberrypi ~ $ ./pj1
Grade:A
pi@raspberrypi ~ $
```

ผลการรัน คือ “ Grade:A ”

ค่าที่แสดงที่หน้าจอจะมาจากค่าในตัวแปร grade ที่ทำการเปลี่ยนค่าแล้ว โดยมีที่มาจาก R0 <- 100 ซึ่งมาจากการบวกกันของ R1 R2 R3 R4 และ R5 ($5+10+20+30+35=100$) จากนั้นทำงานต่อไปที่ A โดยใน A จะเปรียบเทียบค่าใน R0 > 79 ? ถ้าค่าใน R0 > 79 จะทำการรันข้ามไปที่ gradeA ในส่วนนี้จะทำการเปลี่ยนค่าในตัวแปร grade ที่ตำแหน่งที่ 6 ให้เป็น A จาก “Grade: ” จะเป็น “Grade:A ” จากนั้นจะทำงานต่อไปที่ output เพื่อให้ทำการแสดงค่าในตัวแปร grade ที่หน้าจอ

เพิ่มเติม

as -o pj1.o pj1.s	}	คือ ส่วนคำสั่งที่ทำให้โปรแกรมรันไฟล์ที่สร้างไว้และแสดงผลการรันที่หน้าจอ
ld -o pj1 pj1.o		
./pj1		

ผลการทำงานเพิ่มเติม

หาค่าที่เก็บใน R1, R2, R3, R4 และ R5 เป็นค่าอื่น

```
QEMU
CMP R0,#59
BLS Dplus
B gradeC
Dplus:
CMP R0,#54
BLS D
B gradeDplus
D:
CMP R0,#49
BLS gradeF
B gradeD
gradeA:
MOV R0,#0x41
LDH R1,=grade
STHB R0,[R1,#6]
B output
gradeBplus:
MOV R0,#0x42
LDH R1,=grade
STHB R0,[R1,#6]
MOV R0,#0x2B
LDH R1,=grade
STHB R0,[R1,#7]
[ Wrote 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:B+
pi@raspberrypi ~$
```

ผลการรัน “Grade:B+”

เนื่องจาก $74 < R0 \leq 79$

```
QEMU
MOV R1,#5
MOV R2,#10
MOV R3,#20
MOV R4,#30
MOV R5,#5
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
[ Wrote 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:B
pi@raspberrypi ~$
```

ผลการรัน “Grade:B ”

เนื่องจาก $69 < R0 \leq 74$

```
QEMU
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
BLS C
B gradeCplus
C:
CMP R0,#59
BLS Dplus
B gradeC
Dplus:
CMP R0,#54
BLS D
B gradeDplus
D:
CMP R0,#49
BLS gradeF
[ Wrote 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:C+
pi@raspberrypi ~$
```

ผลการรัน “Grade:C+”

เนื่องจาก $64 < R0 \leq 69$

```
QEMU
MOV R1,#5
MOV R2,#10
MOV R3,#10
MOV R4,#15
MOV R5,#20
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
[ Wrote 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:C
pi@raspberrypi ~$
```

ผลการรัน “Grade:C ”

เนื่องจาก $59 < R0 \leq 64$

```
QEMU
MOV R1,#5
MOV R2,#10
MOV R3,#10
MOV R4,#15
MOV R5,#15
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
[ Wrote 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:D+
pi@raspberrypi ~$
```

ผลการรัน “Grade:D+”

เนื่องจาก $54 < R0 \leq 59$

```
QEMU
MOV R1,#5
MOV R2,#10
MOV R3,#10
MOV R4,#15
MOV R5,#10
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
[ Read 99 lines ]
pi@raspberrypi ~$ as -o pj1.o pj1.s
pi@raspberrypi ~$ ld -o pj1 pj1.o
pi@raspberrypi ~$ ./pj1
Grade:D
pi@raspberrypi ~$
```

ผลการรัน “Grade:D ”

เนื่องจาก $49 < R0 \leq 54$

```
QEMU
MOV R1,#5
MOV R2,#10
MOV R3,#10
MOV R4,#15
MOV R5,#5
ADD R6,R1,R2
ADD R7,R3,R4
ADD R8,R5,R6
ADD R0,R7,R8
A:
CMP R0,#79
BLS Bplus
B gradeA
Bplus:
CMP R0,#74
BLS B
B gradeBplus
B:
CMP R0,#69
BLS Cplus
B gradeB
Cplus:
CMP R0,#64
[ Wrote 99 lines ]
pi@raspberrypi ~ $ as -o pj1.o pj1.s
pi@raspberrypi ~ $ ld -o pj1 pj1.o
pi@raspberrypi ~ $ ./pj1
Grade:F
pi@raspberrypi ~ $
```

ผลการรัน “ Grade:F ”

เนื่องจาก $R0 \leq 49$