C Turnin作業2

國立屏東大學 資訊工程系 程式設計(一)

Turnin作業2

- Turnin Code: c.hw2
- Due Date: 2026/03/18 週三 晚上 23:59分截止 **Hard Deadline**

繳交方式說明

本次 Turnin 作業包含多個程式題，建議先為本次 turnin 要繳交的內容建立一個 **外層資料夾**（例如□c.hw2□□切換到該資料夾後再為每一題建立一個 **內層資料夾**（每一題的資料夾名稱已寫於題目前方，例如第一題的資料夾名稱為□p1□□第二題的為□p2□□以此類推），進入到內層資料夾才依照題目要求進行編撰。

同學們可參考如下命令列操作：

```
<ssh 登入系計中後>
[user@ws ~]$ mkdir c.hw2        # 在家目錄建立了一個名為□c.hw2□的資料夾
[user@ws ~]$ cd c.hw2           # 進入□c.hw2□資料夾
[user@ws c.hw2]$ mkdir p1       # 建立一個名為□p1□資料夾
[user@ws c.hw2]$ cd p1          # 切換到□p1□資料夾
[user@ws p1]$ joe swap.c        # 使用 JOE 編輯器對檔名為 swap.c 的檔案進行編輯
```

等到我們完成 p1 的撰寫後，請自行加以編譯與執行程式，確認 **正確無誤** 後回到 **外層資料夾** 使用 turnin▲c.hw2▲p1↵ 指令完成 **繳交第一題的整個資料夾**□

```
[user@ws p1]$ cd ..             # 回到上一層資料夾
[user@ws c.hw2]$ turnin c.hw2 p1 # 使用 turnin 指令提交 p1 的程式碼
```

當然，你也可以等到本次作業要求的所有題目都在 c.hw2 資料夾裡完成後，一次將所有在 **目前資料夾中的所有檔案** 都加以上傳。

假設你已經在 c.hw2 資料夾裡完成所有題目，同時確認檔案的繳交格式正確，並且每個題目的程式檔案皆成功編譯並確認執行結果正確後，我們可以使用以下指令將多餘的（不需要繳交的）檔案加以刪除後，一次將所有檔案繳交：

```
[user@ws c.hw2]$ ls             # 檢視當前資料夾下有哪些內容
p1 p2 p3 p4
[user@ws c.hw2]$ rm -f */a.out  # 移除所有子資料夾中的 a.out 檔案
[user@ws c.hw2]$ turnin c.hw2 . # 使用 turnin 指令繳交該資料夾下的所有內容
```

```
Turning in:
 ./p3/float.h -- ok
 ./p3/float.c -- ok
 ./p2/min.h -- ok
 ./p2/min.c -- ok
 ./p4/round.h -- ok
 ./p4/round.c -- ok
 ./p1/swap.h -- ok
 ./p1/swap.c -- ok
 ./p5/BMI.c -- ok
 ./p5/BMI.h -- ok
 ./p6/grade.c -- ok
 ./p6/grade.h -- ok
 ./p7/point.c -- ok
 ./p7/point.h -- ok
All done.
[user@ws c.hw2]$
```

如果繳交後想要查看已繳交的檔案及相關資訊，可以輸入 `turnin␣-ls␣c.hw2↵` 指令，例如：

```
[user@ws c.hw2]$ turnin -ls c.hw2
.:
total 24
drwxrwx---. 2 turninman turnin 4096 Mar 11 16:32 p1
drwxrwx---. 2 turninman turnin 4096 Mar 11 16:32 p2
drwxrwx---. 2 turninman turnin 4096 Mar 11 16:32 p3
drwxrwx---. 2 turninman turnin 4096 Mar 11 16:32 p4

./p1:
total 8
-rw-rw----. 1 turninman turnin 96 Mar 11 16:32 swap.c
-rw-rw----. 1 turninman turnin 26 Mar 11 16:32 swap.h

./p2:
total 8
-rw-rw----. 1 turninman turnin 104 Mar 11 16:32 min.c
-rw-rw----. 1 turninman turnin  25 Mar 11 16:32 min.h

./p3:
total 8
-rw-rw----. 1 turninman turnin 165 Mar 11 16:32 float.c
-rw-rw----. 1 turninman turnin  74 Mar 11 16:32 float.h

./p4:
total 8
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 round.c
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 round.h

./p5:
```

```
total 8
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 BMI.c
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 BMI.h

./p6:
total 8
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 grade.c
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 grade.h

./p7:
total 8
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 point.c
-rw-rw----. 1 turninman turnin  0 Mar 11 16:32 point.h

[user@ws c.hw2]$
```

本文使用「 」及「 」代表「空白字元」與Enter 換行字元」，並且將使用者輸入的部份使用灰階方式顯示。另外，題目的執行結果中，如果出現「(」、「)」、「:」、「;」、「.」與「,」等符號，皆為英文半形！

本學期作業繳交需要為每一題建立一個資料夾（資料夾名稱為該題題目前方之代號，第一題為p1第二題為p2餘以此類推），繳交方式可參考上述內容，任何未依照正確繳交格式的檔案將以 0 分計。

---

p1 兩數交換函式設計

請參考下面的main.c程式:

```
#include <stdio.h>
#include "swap.h"

int main()
{
   int x, y, *p;
   printf("x=?");
   scanf("%d", &x);
   printf("y=?");
   scanf("%d", &y);
   swap(&x, &y);
```

```
   printf("x=%d y=%d\n", x, y);
}
```

你必須完成名為 **swap.c** 與 **swap.h** 的C語言程式，其中分別包含swap()函式的 **實作** 與其 **原型宣告**swap()函式接收兩個數值的記憶體位址做為參數，並在函式中將這兩個記憶體位址內所包含的數值進行交換。本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c swap.o
        cc main.c swap.o
swap.o: swap.c swap.h
        cc -c swap.c

clean:
        rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
 [3:23 user@ws p1] ./a.out↵
 x=?1↵
 y=?2↵
 x=2▲y=1↵
 [3:23 user@ws p1] ./a.out↵
 x=?3↵
 y=?5↵
 x=5▲y=3↵
 [3:23 user@ws p1]
```

請注意本題應繳交 **swap.c** 及 **swap.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交

---

p2 傳回最小值的記憶體位址函式設計

請參考下面的main.c程式:

```
#include <stdio.h>
#include "min.h"

int main()
{
   int x, y, *p;
```

```
   scanf("%d %d", &x, &y);
   p = min(&x, &y);
   printf("The minimum of %d and %d is %d.\n", x, y, *p);
}
```

你必須完成名為 **min.c** 與 **min.h** 的C語言程式，其中分別包含min()函式的 **實作** 與其 **原型宣告**min()函式接收兩個整數的記憶體位址做為參數，並將其記憶體位址中所包含的數值最小值者的記憶體位址傳回。本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c min.o
        cc main.c min.o

min.o: min.c min.h
        cc -c min.c

clean:
        rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p2] ./a.out↵
1▲2↵
The▲minimum▲of▲1▲and▲2▲is▲1.↵
[3:23 user@ws p2] ./a.out↵
19▲12↵
The▲minimum▲of▲19▲and▲12▲is▲12.↵
[3:23 user@ws p2]
```

請注意本題應繳交 **min.c** 及 **min.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。

---

p3 分解浮點數的函式設計

請參考下面的main.c程式:

```
#include <stdio.h>
#include "float.h"
```

```
int main()
{
   int i_part;
   double f_part;
   double d;
   printf("Please input a floating number: ");
   scanf("%lf", &d);
   decompose(d, &i_part, &f_part);
   printf("i_part=%d\n", i_part);
   printf("f_part=%f\n", f_part);
}
```

你必須完成名為 **float.c** 與 **float.h** 的C語言程式，其中分別包含decompose()函式的 **實作** 與其 **原型宣告**。decompose()函式接收一個浮點數值以及兩個數值的記憶體位址做為參數，並在函式中將所傳入的浮點數的整數與小數部份分別寫入到這兩個記憶體位址內。本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c float.o
        cc main.c float.o
float.o: float.c float.h
        cc -c float.c
clean:
        rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p3] ./a.out↵
Please▲input▲a▲floating▲number:▲3.1415926↵
i_part=3↵
f_part=0.141593↵
[3:23 user@ws p3] ./a.out↵
Please▲input▲a▲floating▲number:▲3↵
i_part=3↵
f_part=0.000000↵
[3:23 user@ws p3] ./a.out↵
Please▲input▲a▲floating▲number:▲0.23↵
i_part=0↵
f_part=0.230000↵
[3:23 user@ws p3]
```

請注意本題應繳交 **float.c** 及 **float.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。

p4 四捨五入函式設計

請撰寫一個函式 **rounding()**，該函式接收一個浮點數位址與保留位數

在設計該函式時必須傳入兩個引數：

- 一個指向該浮點數的指標參數，其浮點數型態為 **double**
- 一個代表需要四捨五入到幾位數的整數參數，其整數型態為 **int**

若輸入數值為 **113.641590** 且 指定四捨五入到 **第三位**，則運算後該位址內容應變更為 **113.642000**

請參考下面的main.c程式，並完成名為 **round.c** 與 **round.h** 的C語言程式，其中分別包含 rounding() 函式的 **實作** 與其 **原型宣告**:

```
#include <stdio.h>
#include "round.h"

int main()
{
    double x;
    int p;
    printf("Please input a floating number: ");
    scanf(" %lf", &x);
    printf("What decimal place do you want to round? ");
    scanf(" %d", &p);
    rounding(&x,p);
    printf("%f\n", x);
}
```

本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c round.o
        cc main.c round.o

round.o: round.c round.h
        cc -c round.c

clean:
        rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p4] ./a.out↵
Please▲input▲a▲floating▲number:▲113.645192↵
What▲decimal▲place▲do▲you▲want▲to▲round?▲0↵
114.000000↵
[3:23 user@ws p4] ./a.out↵
Please▲input▲a▲floating▲number:▲113.645192↵
What▲decimal▲place▲do▲you▲want▲to▲round?▲1↵
113.600000↵
[3:23 user@ws p4] ./a.out↵
Please▲input▲a▲floating▲number:▲113.645192↵
What▲decimal▲place▲do▲you▲want▲to▲round?▲2↵
113.650000↵
[3:23 user@ws p4] ./a.out↵
Please▲input▲a▲floating▲number:▲113.645192↵
What▲decimal▲place▲do▲you▲want▲to▲round?▲3↵
113.645000↵
[3:23 user@ws p4] ./a.out↵
Please▲input▲a▲floating▲number:▲113.645192↵
What▲decimal▲place▲do▲you▲want▲to▲round?▲4↵
113.645200↵
[3:23 user@ws p4]
```

1. 請注意本題應繳交 **round.c** 及 **round.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。
2. 本題四捨五入位數數入範圍不超過 0 $\leq$ n $\leq$ 5

---

p5 BMI計算器

你必須完成名為 **BMI.c** 與 **BMI.h** 的C語言程式。**calculateBMI()** 函式接收體重（weight，單位為公斤）與身高（height，單位為公尺），計算其對應的BMI值。同時，函式必須回傳一個整數值（level）代表健康狀態：

- level 1 = 代表過輕 (BMI < 18.5)
- level 2 = 代表正常 (18.5 <= BMI < 24)
- level 3 = 代表過重 (BMI >= 24)

BMI的計算可參考以下公式： $$ BMI = \frac{weight}{height^2} $$

請參考下面的main.c程式，並完成名為 **BMI.c** 與 **BMI.h** 的C語言程式，其中分別包含 **calculateBMI()** 函式的 **實作** 與其 **原型宣告**:

```
#include <stdio.h>
#include "BMI.h"
```

```
int main()
{
    double weight, height, BMI;
    int level;
    printf("Please input the weight(kg): ");
    scanf(" %lf", &weight);
    printf("Please input the height(m): ");
    scanf(" %lf", &height);
    level = calculateBMI(weight, height, &BMI);
    printf("BMI = %.2f\n", BMI);
    if(level == 1)
        printf("Underweight\n");
    else if(level == 2)
        printf("Normal\n");
    else if(level == 3)
        printf("Overweight\n");
}
```

本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c BMI.o
    cc main.c BMI.o

BMI.o: BMI.c BMI.h
    cc -c BMI.c

clean:
    rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p5] ./a.out↵
Please▲input▲the▲weight(kg):▲70.6↵
Please▲input▲the▲height(m):▲1.52↵
BMI▲=▲30.56↵
Overweight↵
[3:23 user@ws p5] ./a.out↵
Please▲input▲the▲weight(kg):▲50.42↵
Please▲input▲the▲height(m):▲1.71↵
BMI▲=▲17.24↵
Underweight↵
[3:23 user@ws p5] ./a.out↵
Please▲input▲the▲weight(kg):▲70.3↵
Please▲input▲the▲height(m):▲1.78↵
```

```
BMI▲=▲22.19↵
Normal↵
[3:23 user@ws p5]
```

請注意本題應繳交 **BMI.c** 及 **BMI.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。

---

p6 成績加權

請參考下面的main.c程式:

```
#include <stdio.h>
#include "grade.h"

int main()
{
    double score;
    printf("Enter your original score: ");
    scanf("%lf", &score);

    adjustScore(&score);
    printf("Adjusted score: %.2f\n", score);
}
```

你必須完成名為 **grade.c** 與 **grade.h** 的C語言程式，其中分別包含 **adjustScore()** 函式的 **實作** 與其 **原型宣告**。 加權規則如下：

- 將原始成績乘以 1.234。
- 若加權後的成績超過 100 分，則以 100 分計。

本題的相關程式將使用以下的Makefile進行編譯：

```
all: main.c grade.o
    cc main.c grade.o

grade.o: grade.c grade.h
    cc -c grade.c

clean:
    rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p6] ./a.out↵
Enter▲your▲original▲score:▲60↵
Adjusted▲score:▲74.04↵
[3:23 user@ws p6] ./a.out↵
Enter▲your▲original▲score:▲85↵
Adjusted▲score:▲100.00↵
[3:23 user@ws p6] ./a.out↵
Enter▲your▲original▲score:▲0↵
Adjusted▲score:▲0.00↵
[3:23 user@ws p6] ./a.out↵
Enter▲your▲original▲score:▲49↵
Adjusted▲score:▲60.47↵
[3:23 user@ws p6]
```

請注意本題應繳交 **grade.c** 及 **grade.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。

---

p7 二維座標平移

請參考下面的main.c程式:

```
#include <stdio.h>
#include "point.h"

int main()
{
    int x, y, dx, dy;
    printf("Enter current position (x y): ");
    scanf("%d %d", &x, &y);
    printf("Enter displacement (dx dy): ");
    scanf("%d %d", &dx, &dy);

    movePoint(&x, &y, dx, dy);

    printf("New position: (%d, %d)\n", x, y);
}
```

你必須完成名為 **point.c** 與 **point.h** 的C語言程式。 **movePoint()** 函式接收兩個座標軸的記憶體位址□x,

y□□以及要移動的位移量□dx, dy□□函式必須直接更新該座標位址的值，使點移動到新的位置。本題將使用以下的**Makefile**進行編譯：

```
all: main.c point.o
    cc main.c point.o

point.o: point.c point.h
    cc -c point.c

clean:
    rm -f *.o *~ *.*~ a.out
```

此題的執行結果如下：

```
[3:23 user@ws p7] ./a.out↵
Enter▲current▲position▲(x▲y):▲10▲20↵
Enter▲displacement▲(dx▲dy):▲5▲-10↵
New▲position:▲(15,▲10)↵
[3:23 user@ws p7] ./a.out↵
Enter▲current▲position▲(x▲y):▲-5▲3↵
Enter▲displacement▲(dx▲dy):▲7▲-3↵
New▲position:▲(2,▲0)↵
[3:23 user@ws p7] ./a.out↵
Enter▲current▲position▲(x▲y):▲-1▲-5↵
Enter▲displacement▲(dx▲dy):▲-3▲-7↵
New▲position:▲(-4,▲-12)↵
[3:23 user@ws p7]
```

請注意本題應繳交 **point.c** 及 **point.h** 兩個檔案，至於 **main.c** 與 **Makefile** 則不需繳交。