科目名: プログラミング実習III

課題: B2

番号: 27020856

氏名: 趙 聖化

作成: 2023年10月08日

----------------------------------------

[課題 2.1]

課題 (1-4) のプログラム dist.c を書き換え, main 関数から次の関数 dist を呼び出すようにせよ. main と dist は同じファイル内 (dist2.c とする) に記述せよ．

* + double dist(double x1, double y1, double x2, double y2) は, 4 つの double 型の値 x1, y1, x2, y2 を受け取って, 2 点 (x1,y1) と (x2,y2) の距離を double 型で返す関数である.

コンパイルして実行し, 課題 (1-4) と同じ dist-in.txt に対して dist-exp.txt が出力されることを確認せよ.

結果：

gcc dist2.c

chosunghwa@joseongcBookAir 2-1 % ./a.out < dist-in.txt > dist-out.txt

4つの実数 (x1, y1), (x2, y2)を入力して下さい4つの実数 (x1, y1), (x2, y2)を入力して下さい4つの実数 (x1, y1), (x2, y2)を入力して下さい%

chosunghwa@joseongcBookAir 2-1 % diff -w dist-out.txt dist-exp.txt

感想：

関数を作って別の関数で変数を受け取り、出た結果をreturnする方法を学びました。

----------------------------------------

[課題 2.2]

double 型の値 a, b, c (ただし a≠0 とする) を受け取って, ２次方程式 ax2+bx+c=0 の解を標準出力に出力する関数 void qeq(double a, double b, double c) を作成せよ. 解は小数点以下 5 桁で, 1 行に出力せよ. 解が 2 個ある場合には, スペース区切りで小さい順に出力すること. 解がない場合は, 改行のみ出力せよ. なお, 改行の出力は qeq 関数の中で行うこと (main の中で改行を出力しないこと)

この関数を用いることにより, 次の動作を繰り返し行うプログラム qeq.c を作成せよ.

* + 実数 a, b, c を標準入力から入力し, ２次方程式 ax2+bx+c=0 の全ての解を標準出力に出力する.
  + ただし, a=0 の場合には繰り返しを終了する.

結果：

chosunghwa@joseongcBookAir 2-2 % ./a.out < qeq-in.txt > qeq-out.txt

a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : %

chosunghwa@joseongcBookAir 2-2 % diff -w qeq-out.txt qeq-exp.txt

感想：

2方程式の解を出力するときDの値で区分するのを工夫しました。

----------------------------------------

[課題 2.3]

課題 (2-1) で作成したプログラム dist2.c を, 次のファイルに分割し, 分割コンパイルを行え.

* + 関数 main のみを記述した dist3main.c
  + 関数 dist のみを記述した dist3.c
  + 関数 dist のプロトタイプ宣言を記述した dist3.h

結果：

chosunghwa@joseongcBookAir 2-3 % ./a.out < dist-in.txt > dist-out.txt

4つの実数 (x1, y1), (x2, y2)を入力して下さい4つの実数 (x1, y1), (x2, y2)を入力して下さい4つの実数 (x1, y1), (x2, y2)を入力して下さい%

chosunghwa@joseongcBookAir 2-3 % diff -w dist-out.txt dist-exp.txt

感想：

分割コンパイルのをる方法、同じファイルの中で全部関数を作るのではなく、ファイルを分割をする方法を学ぶことができまいた。

----------------------------------------

[課題 2.4]

課題 (2-2) で作成したプログラム qeq.c を, 次のファイルに分割し, 分割コンパイルを行え.

* + 関数 main のみを記述した qeq2main.c
  + 関数 qeq のみを記述した qeq2.c
  + 関数 qeq のプロトタイプ宣言を記述した qeq2.h

結果：

chosunghwa@joseongcBookAir 2-4 % ./a.out < qeq-in.txt > qeq-out.txt

a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : a,b,c : %

chosunghwa@joseongcBookAir 2-4 % diff -w qeq-out.txt qeq-exp.txt

感想：

一つのファイルになっているファイルを分割するのを復習しました。

----------------------------------------

[課題 2.5]

f(x) = ax2 + bx + c に対し, 区間 [x1, x2] における f(x) = 0 の解を, 下記の「二分法」により求めて返す関数

double bisection(double x1, double x2, double a, double b, double c, double eps)

を作成せよ. episilon は計算の許容誤差を決めるパラメータである. ただし, a≠0 であり, f(x1) と f(x2) の符号は異なるものとする

結果：

a, b, c, x1, x2, eps を入力 : 1 0 -2 1.414 5 0.001

i x1 x2 xmid f1 f2 fmid

0 1.41400000 5.00000000 3.20700000 -0.00060400 23.00000000 8.28484900

1.41400000

a, b, c, x1, x2, eps を入力 : -1 0 2 1.414 5 0.0001

i x1 x2 xmid f1 f2 fmid

0 1.41400000 5.00000000 3.20700000 0.00060400 -23.00000000 -8.28484900

0 1.41400000 3.20700000 2.31050000 0.00060400 -8.28484900 -3.33841025

0 1.41400000 2.31050000 1.86225000 0.00060400 -3.33841025 -1.46797506

0 1.41400000 1.86225000 1.63812500 0.00060400 -1.46797506 -0.68345352

0 1.41400000 1.63812500 1.52606250 0.00060400 -0.68345352 -0.32886675

0 1.41400000 1.52606250 1.47003125 0.00060400 -0.32886675 -0.16099188

0 1.41400000 1.47003125 1.44201562 0.00060400 -0.16099188 -0.07940906

0 1.41400000 1.44201562 1.42800781 0.00060400 -0.07940906 -0.03920631

0 1.41400000 1.42800781 1.42100391 0.00060400 -0.03920631 -0.01925210

0 1.41400000 1.42100391 1.41750195 0.00060400 -0.01925210 -0.00931179

0 1.41400000 1.41750195 1.41575098 0.00060400 -0.00931179 -0.00435083

0 1.41400000 1.41575098 1.41487549 0.00060400 -0.00435083 -0.00187265

0 1.41400000 1.41487549 1.41443774 0.00060400 -0.00187265 -0.00063413

0 1.41400000 1.41443774 1.41421887 0.00060400 -0.00063413 -0.00001502

0 1.41400000 1.41421887 1.41410944 0.00060400 -0.00001502 0.00029450

1.41421887

a, b, c, x1, x2, eps を入力 : 1 0 -2 1 2 0.000001

i x1 x2 xmid f1 f2 fmid

0 1.00000000 2.00000000 1.50000000 -1.00000000 2.00000000 0.25000000

0 1.00000000 1.50000000 1.25000000 -1.00000000 0.25000000 -0.43750000

0 1.25000000 1.50000000 1.37500000 -0.43750000 0.25000000 -0.10937500

0 1.37500000 1.50000000 1.43750000 -0.10937500 0.25000000 0.06640625

0 1.37500000 1.43750000 1.40625000 -0.10937500 0.06640625 -0.02246094

0 1.40625000 1.43750000 1.42187500 -0.02246094 0.06640625 0.02172852

0 1.40625000 1.42187500 1.41406250 -0.02246094 0.02172852 -0.00042725

0 1.41406250 1.42187500 1.41796875 -0.00042725 0.02172852 0.01063538

0 1.41406250 1.41796875 1.41601562 -0.00042725 0.01063538 0.00510025

0 1.41406250 1.41601562 1.41503906 -0.00042725 0.00510025 0.00233555

0 1.41406250 1.41503906 1.41455078 -0.00042725 0.00233555 0.00095391

0 1.41406250 1.41455078 1.41430664 -0.00042725 0.00095391 0.00026327

0 1.41406250 1.41430664 1.41418457 -0.00042725 0.00026327 -0.00008200

0 1.41418457 1.41430664 1.41424561 -0.00008200 0.00026327 0.00009063

0 1.41418457 1.41424561 1.41421509 -0.00008200 0.00009063 0.00000431

0 1.41418457 1.41421509 1.41419983 -0.00008200 0.00000431 -0.00003884

0 1.41419983 1.41421509 1.41420746 -0.00003884 0.00000431 -0.00001726

0 1.41420746 1.41421509 1.41421127 -0.00001726 0.00000431 -0.00000647

0 1.41421127 1.41421509 1.41421318 -0.00000647 0.00000431 -0.00000108

0 1.41421318 1.41421509 1.41421413 -0.00000108 0.00000431 0.00000162

1.41421318

感想：

知らない数学公式を見てそこからプログラミングをすることが練習でき、いろんな分岐によってwhile とifを使うのを学びました。

----------------------------------------

[課題 2.6]

* 上記の関数を用いて, 次の動作を繰り返すプログラムを作成せよ.
  + ある人間の名前, 身長, 体重を標準入力から body\_index\_t 型変数 p に読み込む. ただし, 名前が "q" の場合には繰り返しを終了する.
  + 関数 bmi を使って計算した BMI を p.bmi に代入する.
  + 関数 body\_index\_print を使って p の情報を出力する.

結果：

chosunghwa@joseongcBookAir 2-6 % ./a.out < body-in.txt >body-out.txt

chosunghwa@joseongcBookAir 2-6 % diff -w body-out.txt body-exp.txt

感想：

構造体の中の情報を受け取り、他の関数で使う方法を学びました。

----------------------------------------

[課題 2.7]

課題 (2-3) の dist3.c を次のように書き換えて, dist4.c を作成せよ.

* + 点を 2 つの double 型変数ではなく, 次のように宣言された point\_t 型を用いて表わすようにし, 関数 double dist(double x1, double y1, double x2, double y2) を, double dist(point\_t p1, point\_t p2) に書き換えよ.

結果：

chosunghwa@joseongcBookAir 2-7 % ./a.out < dist-in.txt >dist-out.txt

p1 と p2 の座標 (それぞれ x と y) をスペース区切りで : p2 の座標 (x と y): p1 と p2 の座標 (それぞれ x と y) をスペース区切りで : p2 の座標 (x と y): p1 と p2 の座標 (それぞれ x と y) をスペース区切りで : p2 の座標 (x と y): %

chosunghwa@joseongcBookAir 2-7 % diff -w dist-out.txt dist-exp.txt

感想：

.hファイルの中の構造体を入れて他の関数でその構造体の情報を使う方法を学びました。

----------------------------------------

[課題 2.8]

2つの整数 a, b を受け取り, 区間 [a,b] の整数値の乱数を発生させて返す関数 irand(int a, int b) を作成せよ. この関数を用いて次の操作を行うプログラム irand.c を作成せよ.

* + 3つの整数 a, b, n を入力し, irand(a,b) を n 回呼び出して生成された乱数を出力する.
  + a から b までの数について, その数が何回生成されたかを数えて出力する.

表示形式は下記の実行例に倣え.  
次の動作確認を行うこと.

* + a=11, b=20, n=10000 を入力して, 11〜20 がおよそ 1000 づつ生成されていることを確認せよ.
  + a=7, b=8, n=10 を入力して, 7 と 8 の生成数の合計が 10 になっていることを確認せよ.

結果：

A=11, b=20, n=10000텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

A = 7 , b = 8 , n = 10

텍스트, 폰트, 스크린샷, 타이포그래피이(가) 표시된 사진

자동 생성된 설명

感想：区間の整数値の乱数を自分が設定して発生するのと、その値の数を得ることをできるようになりました。

----------------------------------------

[課題 B2 の感想]

今回の授業を通じてc言語で関数を生成し、mainではなく関数から変数を受けて計算してreturnすることを復習することができました。 また、一つのメインファイルではなく、様々なファイルに分けてオブジェクト型にする方法を学び、興味深い部分でした。  
C言語で構造体の使い方を知り、構造体関数をファイルに分けて使う方法も学ぶようになりました。  
  
課題2-5で二分法の関数を作る時一番悩みました