プログラミング I

REPORT#5

提出日	:	2012 年 6 月 28 日　木曜日
所属	:	琉球大学工学部情報工学科
学籍番号	:	125722G
氏名	:	知念 辰明

1.関数の翻訳単位にファイルを分け、同様な動作をするプログラムを作成せよ。

a-0:Makefile の作成。

Makefile の中身▼

```
# RP5のためのmakefileみたいです。

ypf:ypf.o ave.o print_data.o
	gcc -o ypf ypf.o get_ave.o print_data.o

ypf.o:ypf.c
	gcc -c ypf.c

get_ave.o:get_ave.c
	gcc -c get_ave.c

print_data.o:print_data.c
	gcc -c print_data.c
```

解釈▼

・平均値を算出するルーチンを"get_ave.c",各値との差を求め出力するルーチンを"print_data.c",
ルーチンの呼び出しや値の引渡しに使われる main 関数を含んだコア部を"ypf.c",完成後の実行可能ファイルを
"ypf"として Makefile を作成した。
・各種ルーチンに変更があった場合,"make"コマンドを使用するとそれを認識,適当な部分をコンパイル・リンク
してくれる為,ファイルを分けた際に効果を発揮する。

a-1:翻訳単位にファイルを分け,同様な動作をさせる。

main 関数の含まれるファイルの高水準コードの全体▼

```
1   /*
2     program  :ypf.c
3     StudentID:125722G
4     Author   :Tatsunori Chinen
5     date     :2012/06/16 (Sat)
6     comments :show average!
7   */
8   #include <stdio.h>
9
10  int score[10] = {3,5,8,9,10,6,7,9,8,3};    /* int型変数scoreを[0]から[9]まで連続して宣言,値を入力 */
11  int n = 10;                                /* int型整数nを宣言,10という値を代入 */
12  float get_ave();                           /* サブルーチンget_aveを宣言 */
13  void print_data(int n,float ave_top);      /* サブルーチンprint_dataと,引き渡す引数の型を宣言 */
14
15  int main(){
16
17      float ave_top;                         /* ファイル内で有効なfloat型変数ave_topを宣言 */
18      ave_top = get_ave();                   /* 変数ave_topにget_aveが返す値を代入 */
19      printf( "ave = %3.1f\n",ave_top);    /* サブルーチンget_aveが返す値をprintf()関数で出力 */
20      print_data(n,ave_top);                 /* サブルーチンprint_dataを呼び出し,変数nおよびave_topを引渡す */
21      return(0);                             /* プログラムが正常に終了したことを示す */
22
23  }
```

サブルーチン get_ave の高水準コードの一部▼

```
8   float get_ave(){
9
10      int i;                                 /* int型変数iを宣言 */
11      extern int score[10];                  /* mainファイルで宣言されたint型変数score[10]を外部から引き継ぎます */
12      float aver = 0.0;                      /* float型変数averを宣言,"0.0"で初期化します */
13
14      for( i=0 ; i<10 ; i++ ) aver = aver + (float)score[i]; /* 変数averにscoreに格納されている値を全て足す */
15      aver = aver / 10.0;                    /* 10個の数字が足された値であるaverを10で割って平均の値を求める */
16      return(aver);                          /* 変数averに格納されている値をmain関数に返します */
17
18  }
```

サブルーチン print_data の高水準コードの一部▼

```
8   void print_data(int n, float ave_top){
9
10      int i = 0;                             /* int型変数iを宣言 */
11      extern int score[10];                  /* mainファイルで宣言されたint型変数score[10]を外部から引き継ぎます */
12      float dif;                             /* float型変数difを宣言 */
13
14      while( i < n ){                        /* 引き渡されたnの値よりiが小さければ動作を繰り返します */
15          dif = (float)score[i] - ave_top;   /* 変数difにscore[i]に格納されている値とave_topの差を代入 */
16          printf( "score[%02d] = %2d Difference from average = %4.1f\n",i,score[i],dif);
17          /* 現時点でのiの値とscore[i]に格納されている値,求まった差を出力 */
18          i+=1;                              /* iに1を加えます */
19      }
20
21  }
```

実行結果▼

```
ave = 6.8
score[00] =  3 Difference from average = -3.8
score[01] =  5 Difference from average = -1.8
score[02] =  8 Difference from average =  1.2
score[03] =  9 Difference from average =  2.2
score[04] = 10 Difference from average =  3.2
score[05] =  6 Difference from average = -0.8
score[06] =  7 Difference from average =  0.2
score[07] =  9 Difference from average =  2.2
score[08] =  8 Difference from average =  1.2
score[09] =  3 Difference from average = -3.8
```

考察▼

・main 関数の含まれるファイル 10 行目で宣言された int 型変数 score[10]を,サブルーチン get_ave,print_data において記憶域クラス指定子"extern"(外部)を使うことで呼び出している。
・クラス指定子"extern"を使わずにいると,以下のような結果が得られた。

```
ave = -120774400.0
score[00] =  3 Difference from average = 120774400.0
score[01] =  5 Difference from average = 120774408.0
score[02] =  8 Difference from average = 120774408.0
score[03] =  9 Difference from average = 120774408.0
score[04] = 10 Difference from average = 120774408.0
score[05] =  6 Difference from average = 120774408.0
score[06] =  7 Difference from average = 120774408.0
score[07] =  9 Difference from average = 120774408.0
score[08] =  8 Difference from average = 120774408.0
score[09] =  3 Difference from average = 120774400.0
```

・wiki サイト「Open Lecture」におけるサンプルや,"extern"を使った場合とは明らかに違う結果が得られた。
・"extern"を使って呼び出す変数 score[10]と使わずに呼び出す変数 score[10]では異なる内容が格納されている。よって,この 2 つが違う変数であることが分かる。
・"extern"を使うことによって,main 関数のプログラムの外で宣言された score[10]を呼び出している。

・int 型変数 score[10]のスコープ（有効範囲）について,下記のように変数の宣言位置を書き換えたプログラムを作成したところ,エラーが発生した。

main 関数の含まれるファイルの高水準コードの一部▼

```
8   #include <stdio.h>
9
10  int n = 10;                                 /* int型整数nを宣言,10という値を代入 */
11  float get_ave();                            /* サブルーチンget_aveを宣言 */
12  void print_data(int n,float ave_top);       /* サブルーチンprint_dataと,引き渡す引数の型を宣言 */
13
14  int main(){
15
16      int score[10] = {3,5,8,9,10,6,7,9,8,3};/* int型変数scoreを[0]から[9]まで連続して宣言,値を入力 */
17      float ave_top;                          /* ファイル内で有効なfloat型変数ave_topを宣言 */
18      ave_top = get_ave();                    /* 変数ave_topにget_aveが返す値を代入 */
19      printf( "ave = %3.1f\n",ave_top);       /* サブルーチンget_aveが返す値をprintf()関数で出力 */
20      print_data(n,ave_top);                  /* サブルーチンprint_dataを呼び出し,変数nおよびave_topを引渡す */
21      return(0);                              /* プログラムが正常に終了したことを示す */
22
23  }
```

リンクの際のエラー▼

```
Undefined symbols for architecture x86_64:
  "_score", referenced from:
      _get_ave in get_ave.o
      _print_data in print_data.o
ld: symbol(s) not found for architecture x86_64
collect2: ld returned 1 exit status
```

・「score という変数は宣言されていない」というエラーが表示された。
・"make"コマンドおよび makefile を用いた実行ファイルの生成した場合,あるいは"gcc -c"コマンドを各ファイルそれぞれに対して実行し,最後にリンクさせて実行ファイルを作成した場合のいずれも,上記のようなエラーが発生した。
・"make"コマンドを用いなかった場合のリンクの段階で発生することを確認した。
・このことより,main 関数内で宣言した変数のスコープは「ファイル」であり,外のファイルからは呼び出すことができないことが分かった。
・一方で main 関数の外で宣言された変数のスコープは「関数全体」であり,他のファイルからも指定子"extern"を用いることで呼び出すことが出来る。

・ただし,main 関数の外で宣言した変数であっても,クラス指定子"static"が付属していると上記のエラーに類似するエラーが表れた。
・"auto","extern","register","typedef"などの指定子においても,リンクの段階でエラーが表示された。

・これらのことより,main 関数の外で宣言した変数で,かつクラス指定子を用いない場合にのみファイルを超えた変数の取り扱いが出来ることが確認できた。
・int 型変数"n"について同様の試行を行なっても結果は同じであったため,宣言された変数が配列であるかないかによらず同様の挙動をするものである。

a-2:変数の宣言を main 関数の外で全て行う。

main 関数の含まれるファイルの高水準コードの全体▼

```
8   #include <stdio.h>
9
10  int n = 10,i;                           /* int型変数nを10で初期化,iを宣言 */
11  int score[10] = {3,5,8,9,10,6,7,9,8,3};  /* int型変数scoreを[0]から[9]まで連続して宣言,値を入力 */
12  float aver=0.0,dif,ave_top;             /* float型変数averを0.0で初期化,dif,ave_topを宣言 */
13  float get_ave();                        /* サブルーチンget_aveを宣言 */
14  void print_data(int n,float ave_top);   /* サブルーチンprint_dataと,引き渡す変数の型を宣言 */
15
16  int main(){
17
18      printf( "ave = %3.1f\n",get_ave());  /* サブルーチンget_aveが返した値を出力 */
19      ave_top = get_ave();                 /* サブルーチンget_aveが返した値をave_topに代入 */
20      print_data(n,ave_top);               /* サブルーチンprint_dataに変数n,ave_topの値を引渡す */
21      return(0);                           /* プログラムが正常に終了したことを示す */
22
23  }
```

サブルーチン get_ave の高水準コードの一部▼

```
8   float get_ave(){
9
10      extern int i;                       /* 外部変数iを読み込み */
11      extern int score[10];               /* 外部変数scoreを読み込み */
12      extern float aver;                  /* 外部変数averを読み込み */
13
14      for( i=0 ; i<10 ; i++ ) aver = aver + (float)score[i]; /* averにscoreの各値を足す */
15      aver = aver / 10.0;                 /* averを10で割って平均値を求める */
16      return(aver);                       /* averの値をmain関数に返す */
17
18  }
```

サブルーチン print_data の高水準コードの一部▼

```
8   void print_data(int n, float ave_top){
9
10      extern int score[10];               /* 外部変数scoreを読み込み */
11      extern int i;                       /* 外部変数iを読み込み */
12      extern float dif;                   /* 外部変数difを読み込み */
13      i = 0;                              /* iを0で初期化 */
14
15      while( i < n ){
16          dif = (float)score[i] - ave_top;  /* 差を求める */
17          printf( "score[%02d] = %2d Difference from average = %4.1f\n",i,score[i],dif);
18          i+=1;                           /* iに1を足します */
19      }
20
21  }
```

実行結果▼

```
ave = 6.8
score[00] =  3 Difference from average = -3.8
score[01] =  5 Difference from average = -1.8
score[02] =  8 Difference from average =  1.2
score[03] =  9 Difference from average =  2.2
score[04] = 10 Difference from average =  3.2
score[05] =  6 Difference from average = -0.8
score[06] =  7 Difference from average =  0.2
score[07] =  9 Difference from average =  2.2
score[08] =  8 Difference from average =  1.2
score[09] =  3 Difference from average = -3.8
```

考察▼

・クラス指定子"extern"を用いて,関数の内部で変数の宣言を行わないように作成した。
・関数の外で変数を宣言した場合でも,宣言したファイルと変数を使用する関数が同じファイルであれば"extern"を用いずに変数を使用することができた。

a-3:サブルーチンの中にサブルーチンを組み込む。

作成した Makefile▼

```
# 時間変換プログラムのためのめいくふぁいる

timeleep:timeleep.o minute.o hour.o
                        gcc -o timeleep timeleep.o convert_h.o convert_m.o

timeleep.o:timeleep.c
          gcc -c timeleep.c

convert_h.o:convert_h.c
          gcc -c convert_h.c

convert_m.o:convert_m.c
          gcc -c convert_m.c
```

main 関数の含まれるファイルの高水準コードの一部▼

```
8   #include <stdio.h>
9
10  int second = 0;
11  void convert_h(void);
12
13  int main(){
14
15      scanf("%d",&second);
16      if                                      ( second > 60 ){
17          convert_h();
18      } else
19          printf("second = %d\n",second);
20
21      return(0);
22
23  }
```

サブルーチン convert_h が含まれるコードの一部▼

```
8   void convert_m(void);
9
10  void convert_h(void){
11      extern int second;
12      if                                      ( second > 3600 ){
13      int hour;
14      int minute;
15      hour = second / 3600;
16      minute = second % 3600;
17      minute = minute / 60;
18      second = minute % 60;
19      printf( "times = %d hour %d minute %d second\n",hour,minute,second);
20      } else convert_m();
21  }
```

サブルーチン convert_m が含まれるファイルの高水準コードの一部▼

```
8   void convert_m(void){
9       extern int second;
10      int minute;
11      minute = second / 60;
12      second = second % 60;
13      printf( "times = %d minute %d second\n",minute,second);
14  }
```

実行結果▼ ※scanf 関数には"6789"を入力した。

```
6789
times = 1 hour 53 minute 53 second
```

考察▼

・サブルーチンの中にサブルーチンを組み込む構造でプログラムを作成した。
・サブルーチンの中で更にサブルーチンを使う場合,サブルーチンを呼び出すルーチンの外でサブルーチンを宣言することで使用が可能になることが分かった。
・クラス指定子"extern"で,2つ上の階層で宣言された変数を呼び出すことが出来る。

XXX.あとがき。（反省・感想・参考）

a-1.参考サイト・文献。

・『C 実践プログラミング 第三版』（オーム社）
・『Open Lecture』
  http://www.osn.u-ryukyu.ac.jp/lecture/wiki/index.php?Open%20Lecture

・シェルにて man コマンドで得られる各種関数のマニュアル

a-2.反省・感想。

サブルーチンの入れ子に挑戦しました。
案外すんなりと作成することができて拍子抜けです…。
makefile の便利さを噛み締めて,これからのレポートやプログラミングに対応していきたいです！

最後までご覧頂きありがとうございました！