プログラミングＣ 平成 27 年度期末試験

平成 28 年 2 月 5 日

以下の注意事項を守って問題に解答しなさい。

・ 解答開始の指示があるまで、このページ以外を見てはならない

・ この期末試験は102点満点である

・ 問題用紙はこの表紙を含めて両面刷り8枚（全15ページ）である

・ 解答時間は 80 分間とする

・ 退席は開始後 40 分まで認めない

・ 解答用紙は片面刷り3枚である。解答は所定の場所に記述すること

・ 解答用紙の各ページそれぞれに「学籍番号」「氏名」を記入すること

・ 筆記具のみを机の上に置くことができる

・ 机の中には一切の物を入れてはいけない

・ 携帯電話は電源を切り、鞄の中に入れておくこと。着信音がなった場合は不正とみなす

すべての不正行為に対して、断固たる処置を行なう。

問題１（各1点x13=13点）

以下は、テキストファイルを読み込み、その内容を画面に表示するとともに文字数（改行コードなどを含む）と行数をカウントするプログラムである。読み込み対象のテキストファイル名はコマンドライン引数で与える。また、文字数と行数のカウント結果は二番目のコマンドライン引数で与えたファイル名で新たなファイルを作成し、出力することができるが、そうしなかった場合は標準出力に出力する。引数の数が仕様に合わない場合や、ファイルのオープンに失敗した場合はエラーメッセージを標準エラー出力に表示し、プログラムを終了させる。
空欄に適切な語句を入れてプログラムを完成させよ。なお、同じ番号の空欄には同じ内容が入る。

[プログラム]

```
#include <stdio.h>
#include <stdlib.h>

int main(____(1)____ argc, ____(2)____ argv[])
{
  int  ccount = 0, lcount = 0, c;
  ____(3)____ *fpin, *fpout;

  if(____(4)____ == 1 || ____(4)____ > ____(5)____){
    ____(6)____(____(7)____, "!!! Argument Error !!!¥n");
    exit(1);
  }

  if((fpin = ____(8)____) == ____(9)____){
    ____(6)____(____(7)____, "!!! Cannot open input file %s !!!¥n", argv[1]);
    exit(1);
  }

  if(____(4)____ == 2 ) {
     fpout = stdout;
  } else if((fpout = ____(10)____) == ____(9)____){
     ____(6)____(____(7)____, "!!! Cannot open output file %s !!!¥n", argv[2]);
     exit(1);
  }

  while((c = ____(11)____) != EOF){
    ____(12)____;
    ccount++;
    if(c == '¥n') lcount++;
  }

  ____(6)____(fpout, "¥n%d characters¥n",ccount);
  ____(6)____(fpout, "%d lines¥n",lcount);

  ____(13)____(fpin);
  ____(13)____(fpout);
```

```
  return 0;
}
```

[実行例]（%はプロンプト、**太字**はキーボードからの入力を示す（以下の問題でも同じ））
サンプルデータファイルの内容

```
% cat data.txt
The University of Aizu is the first university dedicated to computer science in Japan.
It has about 1,100 students enrolled in its undergraduate and graduate programs.
%
```

引数の数が仕様に合わない場合のエラーメッセージ表示(その他のエラー例は省略)

```
% ./a.out
!!! Argument Error !!!
%
```

読み込み対象のテキストファイル名のみを与えた場合の実行例

```
% ./a.out data.txt
The University of Aizu is the first university dedicated to computer science in Japan.
It has about 1,100 students enrolled in its undergraduate and graduate programs.

168 characters
2 lines
%
```

読み込み対象のテキストファイル名とカウント結果出力先ファイル名を与えた場合の実行例と，カウント結果出力先ファイルの内容確認

```
% ./a.out data.txt output.txt
The University of Aizu is the first university dedicated to computer science in Japan.
It has about 1,100 students enrolled in its undergraduate and graduate programs.
% cat output.txt

168 characters
2 lines
%
```

問題２（各2点x10=20点）

N 個（マクロで 5 と定義してある）の整数データを入力して配列 arr[N]に格納した後、配列中の数値を大きい順に並び換えて表示する（ソート）プログラムがある。詳しいアルゴリズムは以下の通りである。

1．以下の 2,3 のステップを i=0 から N-2 まで順次実行する
2．arr[i]から arr[N-1]の中で最も大きな数値が入っている要素を調べる
3．その要素に格納されている値を arr[i]と入れ替える

空欄に適切な語句を入れてプログラムを完成させよ。なお、同じ番号の空欄には同じ内容が入る。

[プログラム]

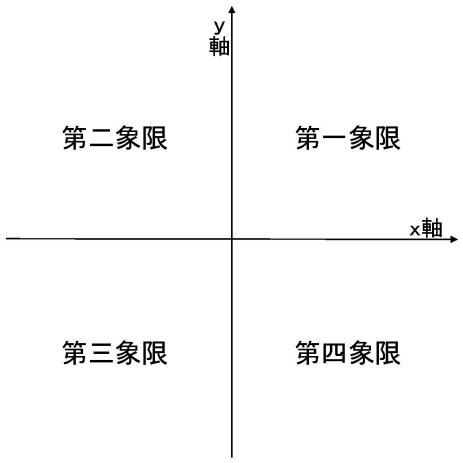
```
#include <stdio.h>
#define N 5

int main()
{
  int arr[N];
  int tmp;
  int *p, *q, *maxptr;

  printf("Please input 5 numbers¥n");
  for(p = ____(1)____; p < ____(2)____; ____(3)____) {
    scanf("%d", p);
  }

  for(p = ____(1)____; ____(4)____; ____(3)____){
    maxptr = ____(5)____;
    for(q = p + 1; q < arr + N; q++){
      if(____(6)____ > ____(7)____) maxptr = q;
    }
    tmp = ____(8)____;
    *p = ____(9)____;
    *maxptr = ____(10)____;
  }

  printf("Sorted data¥n");
  for(p = ____(1)____; p < ____(2)____; ____(3)____) {
    printf("%d¥n", *p);
  }
  return 0;
}
```

[実行例]

```
% ./a.out
Please input 5 numbers
3
2
6
4
1
Sorted data
6
4
3
2
1
%
```

問題３（各1点x20=20点）

平面上の点（簡単のため x 軸上、y 軸上の点は除外する）を考える時、必ず、x 軸、y 軸にて４等分される４つの場所（図のように第１～４象限と呼ぶ）のどこかに位置する。
また、平面上の２点を考えた時、以下のようにそれぞれの点が何象限に位置しているかによって、２点を結ぶ線分が x 軸、y 軸と交差するかを判断する事が出来る。



- ２点が同一象限にある場合は x 軸、y 軸共に交差しない
- ２点が第一と第四象限にある、または第二と第三象限にある場合は x 軸と交差する
- ２点が第一と第二象限にある、または第三と第四象限にある場合は y 軸と交差する
- ２点が第一と第三象限にある、または第二と第四象限にある場合は x 軸 y 軸両方と交差する

二点の座標（x 軸上、y 軸上、原点は除外）を入力し、各々の象限を計算し、更に二点を結ぶ線分が x 軸、y 軸とそれぞれ交差するかどうかを表示するプログラムを作ることを考える。

平面上の１点の x 座標、y 座標とその点が存在する象限を持つ構造体として以下のような xyq 型を宣言する。

```
____(1)____ struct {
  double x; /* x 座標 */
  double y; /* y 座標 */
  int    q; /* 象限(1-4) */
} xyq;
```

プログラムは以下の４つの関数で構成される。
- int main() 各関数を呼び出し、結果を表示する
- xyq *inputxyq(void) xyq 型変数を一つ動的に作成し、scanf で座標を入力し、作成した変数のアドレスを戻り値として返す
- void getquad(xyq *) 一点の構造体変数のアドレスを引数として貰い、その点の位置する象限を構造体メンバーq に入れる
- int iscross(xyq, xyq) 二点を引数として貰い、二点を結ぶ線分が x,y 軸と交差するか判定する（戻り値は 1 の位が x 軸、10 の位が y 軸を表し、1 が交差する、0 が交差しないことを示す。x 軸、y 軸双方と交差する場合は 11、どちらとも交差しない場合は 0）

空欄に適切な語句を入れてプログラムを完成させよ。なお、同じ番号の空欄には同じ内容が入る。また、同じ内容が異なる番号の空欄に入る事もあり得る。

[プログラム]

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

____(1)____ struct {
  double x; /* x 座標 */
```

```
  double y; /* y 座標 */
  int    q; /* 象限(1-4) */
} xyq;

xyq *inputxyq(void); /* 平面上の一点を入力 */
void getquad(xyq *); /* 点の位置する象限を得る */
int iscross(xyq, xyq); /* 二点を結ぶ線分が x,y 軸と交差するか計算する */

int main(){
  xyq *p1,*p2;
  int c;

  /* 二点の作成と座標入力 */
  ____(2)____ = inputxyq();
  ____(3)____ = inputxyq();

  /* 二点の象限を計算 */
  getquad(p1);
  getquad(p2);

  /* x,y 軸との交差調査 */
  c = ____(4)____;

  /* 結果表示 */
  printf("点１：%8.2f %8.2f (%1d 象限)¥n",p1____(5)____,p1____(6)____,p1____(7)____);
  printf("点２：%8.2f %8.2f (%1d 象限)¥n",p2____(5)____,p2____(6)____,p2____(7)____);
  if(c == 0) printf("2 点を結ぶ線分は x 軸、y 軸共に交差しない¥n");
  else{
    if(____(8)____ == 1) printf("2 点を結ぶ線分は x 軸を交差する¥n");
    if(____(9)____ == 1) printf("2 点を結ぶ線分は y 軸を交差する¥n");
  }

  return 0;
}

/* 関数 inputxyq
 * xyq 型構造体を一つ作成し、x,y 座標を入力し、その
 * アドレスを戻り値とする。
 */
xyq *inputxyq(void){
  xyq *p;

  p = (xyq *)____(10)____(sizeof(xyq)); /* xyq 型変数を動的に作成 */
  printf("平面上の一点の座標(x,y)を入力 -> ");
  scanf("%lf%lf",____(11)____, ____(12)____);
  return ____(13)____;
}
```

```
/* 関数 getquad
 * 引数（ポインタ）で指し示される点の象限を調べ、点のメンバ q に書き込む
 */
void getquad(xyq *p){

  int quad; /* 象限を得る変数 */
  if(p->y > 0){
    if(p->x > 0) quad = 1;
    else quad = 2;
  }
  else{ /* p->y < 0 */
    if(p->x < 0) quad = ____(14)____;
    else quad = ____(15)____;
  }

  ____(16)____ = ____(17)____; /* 構造体に象限を書き込む */
}

/* 関数 iscross;
 * 引数の２点の象限から２点を結ぶ線分が x 軸、y 軸と交差するか調べる
 *
 * 戻値  0 ; 両軸とも交差しない（二点が同一象限）
 *       1 : x 軸と交差する（二点が 1,4、2,3 象限）
 *      10 : y 軸と交差する（二点が 1,2、3,4 象限）
 *      11 : 両軸と交差する（二点が 1,3、2,4 象限）
 */
int iscross(____(18)____){

  if(a.q == b.q) return 0; /* 同一象限 */
  if(____(19)____ == 2) return 11; /* 象限が２離れている */
  if(____(20)____) return 1; /* 1,4 象限、2,3 象限 */
  else return 10; /* それ以外、つまり 1,2 象限、3,4 象限 */
}
```

問題４（4点+9点=13点）

問題４－１（各２点 x2=4 点）

以下のプログラム（q4-1.c）を見て問に答えよ。

(1)　何もオプションを付けずにコンパイルした時にインクルードされるのはどのファイルか？（stdio.h を除く）

(2)　プログラムの修正を行わず、コンパイルオプションのみで windows.h がインクルードされるようにするコンパイルコマンドを書け

[プログラム（q4-1.c）]

```
#include <stdio.h>

#if defined UNIX
#include <unix.h>
#elif defined WINDOWS
#include <windows.h>
#else
#include <normal.h>
#endif

int main(){

  return 0;
}
```

問題４－２（各１点 x9=9 点）

マクロを用いて、$f(x,y) = x^2 + y^2$を計算する関数を３種類作成した。以下のプログラムはこれら３つのマクロ関数を使って、f(a,b), f(a+1,b), f(a,b)*2 をそれぞれ求めている。このプログラムの出力結果を予想し、解答用紙の表を埋めなさい。

また、出力結果が数学的に正しい値でない場合はその結果の後ろに×を付けなさい（例：３０　×）。

[プログラム]

```
#include <stdio.h>

#define func1(x,y) x*x+y*y
#define func2(x,y) (x)*(x)+(y)*(y)
#define func3(x,y) ((x)*(x)+(y)*(y))

int main(){
  int a = 2, b = 3;

  printf("¥t(a,b)¥t(a+1,b)¥t(a,b)*2¥n");
  printf("func1 : ");
  printf("%d¥t", func1(a,b));
  printf("%d¥t", func1(a+1,b));
  printf("%d¥n", func1(a,b)*2);
```

```
  printf("func2 : ");
  printf("%d¥t", func2(a,b));
  printf("%d¥t", func2(a+1,b));
  printf("%d¥n", func2(a,b)*2);

  printf("func3 : ");
  printf("%d¥t", func3(a,b));
  printf("%d¥t", func3(a+1,b));
  printf("%d¥n", func3(a,b)*2);
return 0;
}
```

問題５（各2点x8=16点）

n 個の物から k 個取り出す組み合わせの数 ${}_nC_k$ を以下の 3 種類の方法で求めるプログラムを書いた。

方法１：${}_nC_k$ = n*(n-1)*...(n-k+1)/(k*(k-1)*...*2*1) により、ループを使って計算
方法２：${}_nC_k$ = ${}_{n-1}C_{k-1}$ + ${}_{n-1}C_k$ により、関数の再帰呼び出しを使って計算
　　ただし ${}_nC_0$ = 1, ${}_nC_n$ = 1
方法３：まず 0!～n!を計算して配列に格納しておき、${}_nC_k$ = n!/k!/(n-k)! とする
　　ただし 0! = 1

3 種類の方法で計算が行えるように、コメントに従い空欄に適切な語句を入れてプログラムを完成させよ。
計算は 3 つの方法それぞれ別の関数としてある。

[プログラム]

```
#include <stdio.h>
#include <stdlib.h>
#define NMAX 13   /* 階乗の値を入れておく配列のサイズ */

int combi_m(int, int);   /* 方法１ */
int combi_r(int, int);   /* 方法２ */
int combi_f(int, int);   /* 方法３ */
void factarr(int, int *); /* 方法３の中で使用 */

int main(){
  int n, k;
  int status;

  printf("nCk の計算: n と k を入力して下さい: ");
  status = scanf( "%d%d", &n, &k );

  if (status!=2 || n<0 || k<0 || k>n) {
    printf("正しくない入力です。¥n");
    exit (1);
  }
  if (n > NMAX-1) {   /* int 型で計算できる階乗の範囲を越える場合は終了 */
    printf("n が大きすぎます。¥n");
    exit (2);
  }

  printf("方法１: nCk = %d ¥n", combi_m(n,k) );
  printf("方法２: nCk = %d ¥n", combi_r(n,k) );
  printf("方法３: nCk = %d ¥n", combi_f(n,k) );
  return 0;
}

/* 方法１で nCk を求める関数 */
int combi_m(int n, int k){
  int i, r=1;
```

```
  for (i=n ; ____(1)____ ) ____(2)____;
  for (i=k ; ____(3)____ ) ____(4)____;
  return r;
}

/* 方法２で nCk を求める関数 */
int combi_r(int n, int k){

  if (____(5)_____) return 1;
  else return ____(6)____;
}

/* 方法３で nCk を求める関数 */
int combi_f(int n, int k){
  int farr[NMAX];

  factarr( n, farr );   /* 階乗計算関数を呼ぶ */
  return farr[n]/farr[n-k]/farr[k];
}

/* 1 から n の階乗を計算する関数。結果は引数で指し示す配列に格納する。 */
void factarr(____(7)____){
  int i;

  p[0]=1;   /* 0!=1 とする */
  for( i=1; i<=j; i++ ){
    ____ (8)____;
  }
  return;
}
```

[実行例]

```
% ./a.out
nCk の計算: n と k を入力して下さい: 5 3
方法１: nCk = 10
方法２: nCk = 10
方法３: nCk = 10
% ./a.out
nCk の計算: n と k を入力して下さい: 4 0
方法１: nCk = 1
方法２: nCk = 1
方法３: nCk = 1
%
```

問題６（各2点x10=20点）

正の整数をデータとして持ち、データが昇順（小さいものから大きいものへ）に並ぶような連結リストを作成したい。データはキーボードから入力される。入力が正の整数であればリストが昇順を保つように挿入する（例えば、1,3,5,7 というリストの場合、4 は 3 と 5 の間に、10 は最後尾に挿入される）が、既に存在する整数だった場合はエラーメッセージを出力する。入力が負の整数であれば、その絶対値をリストから削除する（例えば、1,3,4,5,7 というリストで-4 が入力されると、リストから 4 を削除する）が、削除すべきデータがリストに無い場合はエラーメッセージを出力する。リストの操作は無限ループで繰り返すものとし、0 が入力されたら終了する。

以下の仕様やコメントに沿うように、空欄に適切な語句を入れてプログラムを完成させよ。なお、同じ番号の空欄には同じ内容が入る。

・ node 構造体へのポインタとして NodePointer 型を定義して使用する
・ 外部変数で定義した head ノードには、データ 0 が入るものとする
・ リスト表示、新規ノード作成、新データ（ノード）追加、削除はそれぞれ関数を作成して使用する

[プログラム]

```
#include <stdio.h>
#include <stdlib.h>

/* node 構造体の宣言、ポインタの宣言 */
struct node{
  int key;
  struct node *next;
};
____(1)____*NodePointer;

/* 関数プロトタイプ宣言 */
void printall();          /* 表示 */
NodePointer make_1node(int, NodePointer); /* 新規ノード作成 */
int inskey(int);   /* 追加（挿入） */
int delkey(int);   /* 削除 */

/* 外部変数 */
NodePointer head;

int main()
{
  int in;
  NodePointer n;

  head = ____(2)____; /* head ノード作成。データは 0 としておく */
  printf( "入力が正の整数なら追加、負の整数ならその絶対値を削除します。¥n" );

  while (1){
    scanf( "%d", &in );    /* キーボード入力を読み込む */

    if ( in == 0 ) break; /* 0 が入力されたら終了。メモリ解放は省略してある */
```

```
    if ( in > 0 ) {   /* 正なら挿入関数を呼ぶ */
      if ( inskey( in ) ) printall();
      else printf("insert error: %d exists in the list!¥n", in );
    }
    else {           /* 負なら符号を逆にして削除関数を呼ぶ */
      if ( delkey( -in ) ) printall();
      else printf("delete error: %d does not exist!¥n", -in );
    }
  }

  return 0;
}

/* 表示関数：(head ノード以外の) リスト全体を表示する */
void printall(){
  NodePointer n;

  for (____(3)____; n=n->next ){
    printf( "< %d ", n->key );
  }
  printf( "¥n" );
}

/* 新規ノード作成関数：引数はデータの整数、次ノードへのポインタ */
NodePointer make_1node(int keydata, NodePointer n){
  NodePointer newnode;

  newnode = (NodePointer) malloc(____(4)____);

  newnode->key = keydata;
  newnode->next = n;
  return newnode;
}

/* 挿入関数：引数のデータが既にリストに存在する場合、0 を返す。
   新しいデータなら、リストが昇順に並ぶように新ノードを追加し、1 を返す */
int inskey(int keydata){
  NodePointer n, nnew;

  for ( n=head; n->next!=NULL; n=n->next ){
    if (____(5)____ == keydata) return 0;   /* 同じ整数がリストに有った場合 */
    else if (____(5)____ > keydata) break;
  }

  nnew = make_1node( keydata, n->next ); /* 新規作成 */
  ____(6)____;   /* 新ノードをリストに接続 */
  return ____(7)____;
```

```
}

/* 削除関数：正の整数を与えて該当するノードを削除し、戻り値 1 を返す。
   該当するものが存在しない場合は、0 を返す。 */
int delkey(int keydata){
  NodePointer n, ndel=NULL;
  /* ndel は初期値を NULL としておき、削除対象のノードが見つかれば、それを ndel とする */

  for ( n=head; n->next!=NULL; n=n->next ){
    if (____(5)____ > keydata) break;    /* keydata を通り過ぎたらループ脱出 */
    if (____(5)____ == keydata) {   /* 見つかった場合 */
      ndel=____(8)____;
      break;
    }
  }

  if (____(9)____) return 0;   /* 見つからなかった場合は 0 をリターン */
  else {
    n->next = ____(10)____;   /* ノードのつなぎ替え */
    free( ndel );
    return 1;
  }
}
```

[実行例]

```
% ./a.out
入力が正の整数なら追加、負の整数ならその絶対値を削除します。
3
< 3
5
< 3 < 5
4
< 3 < 4 < 5
9
< 3 < 4 < 5 < 9
-4
< 3 < 5 < 9
5
insert error: 5 exists in the list!
6
< 3 < 5 < 6 < 9
-1
delete error: 1 does not exist!
0
%
```