プログラミングＣ　2020 年度期末試験

2020 年 12 月 3 日 13:30-15:00　(90 分)

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

・問題用紙はこの表紙を含めて両面刷り９枚（全１８ページ）である

・途中での提出・退席は 14 時 10 分まで認めない

・解答用紙は片面刷り 4 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

・携帯電話等は電源を切り、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

・問題プログラム中で、例えば「`exit(2);`」の「`(2)`」のようなカッコ書き数字は、

問題番号ではないので注意すること

・フォントの都合上、バックスラッシュ「＼」は「¥」として印刷されている

・実行例ではキーボード入力をイタリックで表示してある

・本試験は 120 点満点である

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（20 点）

問題１－１

次に示すコードは、入力ファイルから内容を一文字ずつ読み取り、出力ファイルにそのまま書き出すとともに、数字／英字／改行／それ以外（改行以外の制御文字や記号など）の数を数えて、結果を最後にディスプレイに表示し、行ごとの文字数（改行コードは含めず数える）の最大値もあわせて表示するプログラムである。
文字の種類の判定と、一行あたりの最大文字数カウントは、clschar 関数を作成して用いる。この関数は、char 型の引数一つをとり、改行コードが出現するまで文字数をカウントするほか、文字の種類に応じて 0（数字），1（英字），2（改行），3（その他のとき）の戻り値を返す。
プログラムが上記の動作をするように、空欄を埋めなさい。ただし、同じ番号の空欄には同じものが入る。なお、入力ファイル名と出力ファイル名はコマンドラインオプションでこの順に与えるものとし、オプションの数が合わないときは、標準エラー出力にメッセージを表示して終了する。また指定されたファイルが開けないときも、標準エラー出力にメッセージを表示して終了するものとする。

[プログラム]

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

int clschar(char);

int lmax = 0;   /* 行の最大長さ */

int main(int argc, ___(1)___){
  char cin;
  int cnt[4] = {0, 0, 0, 0}, n;
  FILE *fpin, *fpout;

  if (argc != 3){
    fprintf( ___(2)___, "Usage: ./a.out inputfile outputfile¥n" );
    exit (1);
  }

  /* 入力用ファイルオープン */
  fpin = fopen( _____(3)_____ );
  if (fpin == NULL){
    fprintf( ___(2)___, "ERROR in opening input file!¥n" );
    exit (2);
  }
  /* 出力用ファイルオープン */
  fpout = fopen( _____(4)_____ );
  if (fpout == NULL){
    fprintf( ___(2)___, "ERROR in opening output file!¥n" );
    fclose( fpin );
    exit (2);
  }
```

次ページに続く

```
  /* 入力とカウント、出力 */
  while ((cin = _____(5)_____) != EOF){
    n = clschar( cin );   /* 文字の種類を判定する関数を呼ぶ */
    cnt[n]++;
    fputc( _____(6)_____ );
  }

  /* 統計情報の表示 */
  /* 数字・英字・改行・それ以外の字数と、行の長さの最大値 */
  printf( "--------------------¥n" );
  printf( "numbers: %d, alphabets: %d¥n", cnt[0], cnt[1] );
  printf( "newlines: %d, others: %d¥n", cnt[2], cnt[3] );
  printf( "maximum line width: %d¥n", lmax );

  /* ファイルの後処理 */
  fclose( fpin );
  fclose( fpout );
  return 0;
}

/* clschar 関数：一行の最大文字数をカウントするとともに、文字の種類を判定する。
   戻り値は、0（数字）, 1（英字）, 2（改行）, 3（その他）とする */
int clschar(char c){
  static int nc=0;   /* nc: 一行文字数 */

  nc++;         /* 文字数カウント */

  if (_____(7)_____) return 0;
  else if (_____(8)_____) return 1;
  else if (c == '¥n') {
    nc--;       /* 改行コードは一行文字数に含めない */
    if (___(9)___) ___(10)___;   /* 行の長さが最大かチェック */
    ___(11)___;     /* 一行文字数をリセット */
    ___(12)___;
  }
  else return 3;
}
```

[実行例]

```
% ./a.out UoA.txt output.txt
numbers: 4, alphabets: 24
newlines: 2, others: 6
maximum line width: 22
```

次ページに続く

[入力ファイル UoA.txt の中身]（同じ内容が output.txt にも出力される）

```
The University of Aizu
(since 1993)
```

問題１－２

問題１－１のプログラムに出てきた変数のうち、（ア）lmax、（イ）cin、（ウ）c、（エ）nc について、以下の文章に当てはまるものをすべて、ア～エの記号で答えなさい。

（１）自動変数

（２）宣言されてからプログラム終了まで消滅しない変数（main 関数の自動変数は除く）

（３）通用範囲が変数宣言のある関数の内部だけである変数

（４）プログラム中で一回だけ初期化されている変数

問題２（20 点）

以下のプログラムの出力を答えなさい（"(1)" などの問題番号部分は改めて書かなくてよい)。

```
#include <stdio.h>
#include <string.h>
#define  N 15
int main(){

  char data1[]="The University of Aizu";
  char data2[][N] = {"Aizuwakamatsu","Iwaki","Shirakawa","Koriyama","Fukushima"};/* 都市 */
  char *data3[] = {"Abukuma", "Agano", "Tadami"};  /* 川 */
  int pd[] = {309, 274, 194, 436, 372};  /* 人口密度 */
  char **q;

  int *p, i, j = 0;
  printf( "(1) %d¥n", (int) sizeof(data1) );
  printf( "(2) %d¥n", (int) sizeof(data2) );
  printf( "(3) %d¥n", (int) sizeof(data3) );

  for (i=0; i<2; i++) j += strlen(data3[i]);
  printf( "(4) %d¥n", j );

  strcpy(&data1[4],data3[1]);

  printf( "(5) %s¥n", &data1[1] );
  printf( "(6) %c¥n", *data1-1 );
  printf( "(7) %s¥n", data1+4 );

  p = pd;
  printf( "(8) %d¥n", *(p+3)-*p+1 );

  q = &data3[1];
  printf( "(9) %s¥n", (*q)++ );
  printf( "(10) %s¥n", *(--q)+1);

  return 0;
}
```

問題３（24 点）

問題３－１

以下のプログラム prog03a.c は、ヘッダファイル prog03.h で定義された Record 型構造体変数 data に、input_data 関数を使ってキーボードから学籍番号・科目数分の成績（マクロ SUBJECT_NUM で科目数が定義されている）を入力した後、calc_average 関数で全科目の平均点を計算して構造体変数 data 中のメンバ average に格納し、print_data 関数で data の内容を平均点も含めて出力するものである。空欄を埋めてプログラムを完成させよ。

[プログラム prog03.h]

```
#define SUBJECT_NUM 3 /* 科目数 */
typedef struct{
        char id[9];    /* 学籍番号 (s12xxxxx) */
        int score[SUBJECT_NUM]; /* 成績（科目数分） */
        double average; /* 成績平均点 */
}Record;
```

[プログラム prog03a.c]

```
#include <stdio.h>
#include "prog03.h"

Record input_data(void);
void   calc_average(Record *);
void   print_data(Record);

int main(){
  Record data;
  int i;

  ___(1)___ = input_data();
  calc_average(___(2)___);
  print_data(___(3)___);

  return 0;
}

Record input_data(void) {
  ___(4)___ data;
  int i;
```

次ページに続く

```
  /* 標準入力（キーボードから）のデータ入力 */
  printf("データを入力して下さい¥n");
  printf("学籍番号 (s12xxxxx) -> ");
  scanf("%s", ___(5)___);
  printf("成績（科目数分） -> ");
  for (i = 0; i < SUBJECT_NUM; i++) {
    scanf("%d", ___(6)___);
  }

  return ___(7)___;
}

void   calc_average(Record *data_ptr){
  double sum = 0.0;
  int i;

  for (i = 0; i < SUBJECT_NUM; i++) {
    sum += ___(8)___;
  }

  ___(9)___ = sum/SUBJECT_NUM;
}

void   print_data(Record data){
  int i;

  /* データ表示 */
  printf("¥n 学籍番号: %s¥n", ___(10)___);
  printf("成績:¥n");
  for (i = 0; i < SUBJECT_NUM; i++) {
    printf("  科目%d: %d¥n", i+1, ___(11)___);
  }
  printf("  平均: %f¥n", ___(12)___);
}
```

[実行例]（斜体字はキーボードからの入力）

% ./a.out
データを入力して下さい
学籍番号 (s12xxxxx) -> *s1290001*
成績（科目数分） -> *90 60 80*

次ページに続く

```
学籍番号: s1290001
成績:
  科目 1: 90
  科目 2: 60
  科目 3: 80
  平均: 76.666667
%
```

問題３－２

以下のプログラム prog03b.c は prog03a.c を拡張し、複数人の情報を Record 型構造体変数の配列（マクロ MAX_NUMBER で要素数が定義されている）data_arr に格納するようにしている．入力には新たに input_data2 関数を用いる。input_data2 関数は Record 型構造体変数の配列（ポインタ）を引数とし、関数内で配列の各要素に順次学籍番号・科目数分の成績を入力するようにしている。また、入力が行われた回数（人数）をカウントし、戻り値としている。この変更に伴って、平均点の計算と内容の出力も data_arr の入力済みの全要素に対して行うようにするが、calc_average 関数と print_data 関数、ヘッダファイル prog03.h は変更せずそのまま使用する。空欄を埋めてプログラムを完成させよ。

[プログラム prog03b.c]

```
#include <stdio.h>
#include "prog03.h"
#define MAX_NUMBER 20 /* 人数 */

int    input_data2(Record *);
void   calc_average(Record *);
void   print_data(Record);

int main(){
  Record data_arr[MAX_NUMBER];
  int i, people_num;

  people_num = input_data2(___(13)___);

  for(i = 0; i < people_num; i++) {
    calc_average(___(14)___);
  }

  for(i = 0; i < people_num; i++) {
    print_data(___(15)___);
  }
  return 0;
}
```

次ページに続く

```
int input_data2(Record *data_ptr) {
  int i, j, result;

  /* 標準入力（キーボードから）のデータ入力 */
  for (i = 0; i < MAX_NUMBER; i++) {
    if (scanf("%s", ___(16)___) == EOF) return ___(17)___;
    for (j = 0; j < SUBJECT_NUM; j++) {
      scanf("%d", ___(18)___);
    }
  }
  return MAX_NUMBER;
}

void   calc_average(Record *data_ptr){
  /* prog03a.c から変更なし */
}

void   print_data (Record data){
  /* prog03a.c から変更なし */
}
```

[実行例]（斜体字はキーボードからの入力）

```
% ./a.out
s1290001 90 60 80
s1290002 50 40 60
Ctrl+D
学籍番号: s1290001
成績:
  科目 1: 90
  科目 2: 60
  科目 3: 80
  平均: 76.666667

学籍番号: s1290002
成績:
  科目 1: 50
  科目 2: 40
  科目 3: 60
  平均: 50.000000
%
```

問題４（16 点）

問題４－１

以下の空欄に当てはまる語句を書け。なお、同じ番号の空欄には同じ語句が入る。

分割コンパイルは複数人での開発に用いられる手法である。
プログラムを分割して開発する時に大事なのは、各自の技術に見合った分割量の設計と、関数相互のやりとり、つまり（　1　）の設計である。共通の（　2　）をインクルードする事は、関数の引数の（　3　）、（　4　）、戻り値の（　3　）を開発メンバー間で共有するためにも非常に大切である。その他にも（　2　）には、構造体の（　5　）、マクロ定義などが含まれる。
プログラムのコーディングが終了すると、まず行われるのが（　6　）テストである。これは各開発単位ごと（通常はいくつかの関数を含むファイルごと）に行われるもので、そのファイルに含まれていない関数を仮にシミュレートするような機能を付け加えて、独立して動作できるようにするものである。
各ファイルの（　6　）テストが終了したら、（　7　）テストが行われる。これは分割された全てのファイルを（　7　）し、全ての機能を確認するものである。

問題４－２

以下のファイルは分割コンパイルのテストのために作られた簡単なプログラムを３つの部分に分けたファイルである。
このプログラムはコマンドライン引数から２つの点の平面座標 `(x, y)` を入力し、原点と各点を結ぶ直線（以後、その点の「ベクトル」と呼ぶ）と x 軸との角度（単位：度）を計算し、さらに二つのベクトルのなす角度を計算するものである。その点のベクトルと x 軸との角度は数学関数 `double atan2(double y, double x)` を用いて計算する。なお、`atan2(y,x)` は `y/x` の逆正接、つまり $\tan^{-1}(y/x)$ を返す関数であり、戻り値の範囲は $-\pi$ から $+\pi$ まで、単位はラジアンである。（`x=0` の場合も `y` の正負に応じて $+\pi/2$ か $-\pi/2$ を返す）

(1) `q4a.c` の（問１の語句６）テストのためのコンパイルコマンド、実行コマンドと実行結果を書け。ただし、二点の座標は `(3,3), (-3,3)` とし、この順に入力する。

(2) `q4b.c` を（問１の語句６）テストのためコンパイルしたとする。生成した実行ファイルを実行したときの結果を書け。

(3) `q4a.c` と `q4b.c` の（問１の語句７）テストのためのコンパイルコマンドと実行結果を書け。なお、コンパイルは、個々にソースファイルをコンパイルし、最後にそれらを一つにした実行ファイルを作る分割コンパイルの手法を取ることとし、そのための命令（コマンド）をすべて書きなさい。また、実行時に与える二点の座標は `(2,2), (0,1)` とし、この順に入力されたものとする。

[プログラム]

[q4.h]

```
struct xyd {
  int x;
  int y;
  double theta;
};

void calcang(struct xyd *);
void dispxyd(struct xyd);
```

次ページに続く

[q4a.c]

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "q4.h"

int main(int argc, char *argv[]){
  struct xyd a,b;
  int i;

#ifdef DEBUG
  for(i = 1; i< argc; i++)
    printf("[%d] %s¥n",i, argv[i]);
#endif

  a.x = atoi(argv[1]); a.y = atoi(argv[2]);
  b.x = atoi(argv[3]); b.y = atoi(argv[4]);

#ifdef DEBUG
  a.theta = 40.0; b.theta = 100.0;
#else
  calcang(&a);
  calcang(&b);
  dispxyd(a);
  dispxyd(b);
#endif

  printf("angle between 2 points is %.2f¥n", b.theta - a.theta);

  return 0;
}
```

[q4b.c]

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "q4.h"

#ifdef DEBUG
int main(){
  struct xyd a = {0, 1};
  calcang(&a);
  dispxyd(a);
  return 0;
}
#endif
```

次ページに続く

```
void calcang(struct xyd *p){

  p->theta = atan2((double)p->y, (double)p->x) * 180/M_PI;
  /* 注：M_PIは円周率（math.hで定義されている） */

}

void dispxyd(struct xyd c){

  printf("(%d, %d) angle : %.2f¥n", c.x, c.y, c.theta);

}
```

問題５（18 点）

２つの自然数の最大公約数を求める手法の一つとして、ユークリッドの互除法がある。

ユークリッドの互除法とは、２つの自然数 a, b (a ≧ b) について、a を b で割った際の剰余を r とすると、「aとbの最大公約数」は「bとrの最大公約数」に等しい、という性質を利用して繰り返し計算を行い、剰余が 0 になった時の除数を「aとbの最大公約数」として求める方法である。

最大公約数を求める関数を gcb とすると、負でない整数 a, b (a ≧ b) に対して、

```
gcd(a, b) = gcd(b, r)   (b > 0 のとき)
          = a           (b = 0 のとき)
```

のように、再帰を利用してユークリッドの互除法を定義できる。

この説明に基づいて、以下のプログラムのコメントを参考に、(1)～(3)を埋めて、プログラムを完成させなさい。ただし、このプログラムでは、最大公約数を求める際に関数 gcb を呼び出した回数を記憶するために、整数型変数 count を外部変数として宣言し、使用している。

さらに、main 関数の各 printf 文における(a)～(f)に当てはまる実行結果を答えなさい。(出力フォーマット指定子 %d で出力される値のみ答えれば良い。)

[プログラム]

```
#include <stdio.h>

int gcb(int, int);

int count; /* 関数の呼び出し回数を記憶する外部変数 */

int main() {
    count = 0;
    printf("(a) gcb(16, 12): %d¥n", gcb(16, 12));
    printf("(b) count: %d¥n", count);

    count = 0;
    printf("(c) gcb(59, 27): %d¥n", gcb(59, 27));
    printf("(d) count: %d¥n", count);

    count = 0;
    printf("(e) gcb(333, 185): %d¥n", gcb(333, 185));
    printf("(f) count: %d¥n", count);

    return 0;
}
```

次ページに続く

```
/*
 * 再帰で最大公約数を求める（ユークリッドの互除法）
 * a, b (a ≧ b) が負でない整数なのは保証されているものとする
 */
int gcb(int a, int b){

    count++; /* gcb 関数の呼び出し回数をカウント */

    if(_____(1)_____){ /* 再帰終了条件 */
        _____(2)_____;
    }
    else{ /* 再帰条件 */
        _____(3)_____;
    }
}
```

問題 6 （22 点）

以下のプログラムは、学生の成績データを連結リストで格納し、最高点をとった学生情報を表示するプログラムである。連結リストの構造は第 9 回のハンドアウトと同様であるが、int 型構造体メンバ key の代わりに学籍番号・名前・成績からなる Student 型構造体を使用する。

以下の仕様および実行例を参考に、空欄を埋め完成させなさい。ただし、アロー演算子を使えるところでは、優先して使用すること。なお，ReadFile 関数、finditem 関数、listprint 関数の定義は省略している。

仕様：連結リストは一方向で、先頭に外部変数として宣言された head から始まり、終端は NULL になっているものとする。まず head ノードを作成し、次に ReadFile()関数内でファイルからデータを読み込み、初期リストを作成する。その後、キーボードから学生データを読み込み、リストの最後に追加していく。もし、入力された学生データの学籍番号が既にリスト内に存在していた場合は、入力された新しいデータに置き換えるものとする。ctrl+d でデータの入力は終了し、リスト中の最高得点の学生データを表示する。

append 関数：学生データをリストの最後に追加する。ただし、リスト内に同じ学籍番号があった場合は、古いデータと新しいデータを置き換える。置き換えの具体的な処理は、新しいノードを作り、古いノードと入れ換え、古いノードを削除することとする。

maxnode 関数：リスト中の最高得点のノードを返す。最高得点者が複数いた場合は、リスト中で先頭に近い者のノードを返す。

printnode 関数：引数で与えられたノード内の学生データを表示する。

listprint 関数：リストを表示する。

make_1node 関数: ノードを 1 つ作成する。

finditem 関数：引数で与えられた学籍番号（int 型）と一致するノードの 1 つ前のノードポインタを返す。一致するノードがない場合は NULL を返す。

[実行例]（斜体字はキーボードからの入力）

```
% ./a.out
[Initial]
Head -
 1231001 Honda          83
 1231002 Miura          90
 1231004 Nakamura       60
 1231010 Okazaki        81
 1231012 Nakayama       95
Append new data: (ID name score) -> 1231011 Kubo 88
Head -
 1231001 Honda          83
 1231002 Miura          90
 1231004 Nakamura       60
 1231010 Okazaki        81
 1231012 Nakayama       95
 1231011 Kubo           88
```

次ページに続く

```
Append new data: (ID name score) -> 1231004 Nakata 75
Head -
 1231001 Honda         83
 1231002 Miura         90
 1231004 Nakata        75
 1231010 Okazaki       81
 1231012 Nakayama      95
 1231011 Kubo          88
Append new data: (ID name score) -> ^D
MAX :  1231012 Nakayama      95
%
```

[プログラム]

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

typedef struct {
  int id;
  char name[15];
  int score;
} Student;

typedef struct node *NodePointer;

struct node {
  Student stu;
  struct node *next;
};

NodePointer finditem(int);
NodePointer make_1node(Student, NodePointer);
NodePointer append(Student);
NodePointer maxnode(void);
void listprint(void);
void printnode(NodePointer);
void ReadFile(void);

NodePointer head;

int main()
{
  Student stu;

  printf("[Initial]¥n");
  head = make_1node(_____(1)_______); /* headの初期化（作成） */

  ReadFile();  /* ファイルから読み込み、初期リストを作成 */
  listprint();

  while (1) {
    printf("Append new data: (ID name score) -> ");
    if (scanf("%d%s%d", _________________(2)_________________) ==EOF) break;
    append(stu); /* 学生データの追加 */
    listprint();
  }
```

次ページに続く

```
  printf("¥nMAX : ");
  printnode(maxnode()); /* 最高点を持つノードを表示 */
  printf("¥n");
  return 0;
}

NodePointer make_1node(Student stu, NodePointer p)
{
  NodePointer n;

  if ((n = (NodePointer)malloc(sizeof(struct node))) == NULL) {
    printf("Error in memory allocation¥n");
    exit(8);
  }
  n->stu = stu;
  n->next = p;

  return n;
}

NodePointer maxnode(void)
{
  NodePointer n,maxn;

  maxn=head->next;
  for (n=_______(3)________; n!=NULL ; n=n->next) {
    if (_______________(4)_______________) maxn=n;
  }
  return maxn;
}

void printnode(NodePointer n)
{
  printf(" %d %-14s %d¥n", ____________________(5)_________________);
}

NodePointer append(Student d)
{
  NodePointer n,newnode,delnode;

  if ((n=finditem(d.id))==NULL)  { /* 追加処理 */
    ____________________(6)_____________________ ; /* 最後のノードを探す */
    newnode=make_1node(d,____(7)_____);
    _____(8)_____=newnode;
  } else { /* 置換処理 */
    delnode=_____(9)_____ ;
    newnode=make_1node(d,_______(10)________);
    _____(11)_____=newnode;
    free(delnode);
  }
  return newnode;
}

/* finditem関数、listprint関数、ReadFile関数の定義は省略 */
```