プログラミング基礎 前期期末試験

全てのプログラムファイルの先頭行に、C のコメントとして自分の番号と名前を書くこと。

1「16 進数で入力された整数に対する 10 進数値を出力する処理を繰り返す」プログラムを作ることを考える。
この際、次のような動作となるようにしたい。

- 入力された 16 進数値が「0（つまり整数の 0)」の場合は、その繰り返し処理を終了する
- 入力された 16 進数値が、「0x0〜0xffff」の範囲内であれば、その値を 10 進数で出力し、それ以外は「範囲外である」ことを出力する

上記の動作をするプログラムの一部が、以下のようになった場合、不足している部分を追加して、プログラムを完成させなさい。

```
int main(void)
{
    int num;
                /* ← ここで num に初期値を入れる */
    while(num != 0) {
        printf("16 進数 > ");
        scanf("%x", &num);
        /* ↓ ここに num の値に対する if 文を作る */

    }
    return 0;
}
```

このプログラムの実行例は以下のようになる。

```
[実行結果]
  16 進数 > A5
  10 進数では 165
  16 進数 > FFFF
  10 進数では 65535
  16 進数 > 10000
  out of range
  16 進数 > 0
  out of range
```

2 引数で指定された文字 ch と整数 n に対して、「ch から ch+n までの文字とその ASCII コードを 16 進数で出力する」関数 show_ascii() を作成しなさい。
この関数のプロトタイプ宣言は以下のようになる。

```
void show_ascii(char ch, int n);
  /* 繰り返し処理を作り、 */
  /* ch+0 から ch+n までの文字と 16 進数の整数を出力する */
```

main() での動作確認の例とその実行結果は以下のようになる。

```
[main() での処理]
  show_ascii('a', 3);   /* 'a' から'a'+3 まで出力 */
  show_ascii('Y', 10);  /* 'Y' から'Y'+10 まで出力 */
[実行結果]
```

```
  a(61) b(62) c(63) d(64)
  Y(59) Z(5a) [(5b) \(5c) ](5d) ^(5e) _(5f) `(60) a(61) b(62) c(63)
```

3 引数 ch に対して、「'0'〜'9' の文字を 1 桁の整数値に変換する」関数 to_int() を作成しなさい。ただし、変換処理自体には **atoi() などの標準ライブラリ関数は使わないこと。**

この関数のプロトタイプ宣言は以下のようになる。

```
int to_int(int ch);
  /* ch が'0'〜'9' の範囲内かどうかを調べる */
  /* 範囲内の場合、ch から'0' 引くと整数値になり、この値を戻り値とする */
  /* それ以外の場合、整数値 -1 を戻り値とする */
```

main() での動作確認の例とその実行結果は以下のようになる。

```
[main() での処理]
  printf("%d\n", to_int('0'));  /* ← '0'〜'9' の範囲内の場合 */
  printf("%d\n", to_int('9'));
  printf("%d\n", to_int('/'));  /* ← 範囲外の場合 */
  printf("%d\n", to_int(':'));
[実行結果]
  0
  9
  -1
  -1
```

4 整数値を格納している 2 次元配列 numbers を以下のようにグローバル変数として宣言する。

```
#define ROW 5  /* 行 (row)    */
#define COL 3  /* 列 (column) */
int numbers[COL][ROW] = {
    {6, 1, 3, 10, 6},
    {5, 9, 6,  1, 7},
    {6, 9, 8,  2, 4}
};
```

この 2 次元配列 numbers の指定した行 r に対して、「指定した比較演算で、引数で指定した整数 x との比較結果が true (真) となる個数を求める」関数 count_row() を作成しなさい。

この関数のプロトタイプ宣言は以下のようになる。

```
int count_row(int r, char cond, int x);
  /* 引数 r の行に対する繰り返し処理を作る */
  /* 引数 cond の文字によって、次のような比較演算の処理に分ける */
  /*  ・cond が 'e' の場合、引数 x の値と等しい要素数の個数を数える */
  /*  ・cond が 'l' の場合、引数 x の値より小さい要素数の個数を数える */
  /*  ・cond が 'g' の場合、引数 x の値より大きい要素数の個数を数える */
```

main() での動作確認の例とその実行結果は以下のようになる。

```
[main() での処理]
  printf("%d\n", count_row(0, 'e', 6));  /* 0 行目において 6 と等しい要素数をもとめる */
  printf("%d\n", count_row(1, 'l', 7));  /* 1 行目において 7 より小さい要素数をもとめる */
  printf("%d\n", count_row(2, 'g', 4));  /* 0 行目において 4 より大きい要素数をもとめる */
[実行結果]
  2
  3
  3
```

5 前問で宣言した 2 次元配列 numbers において、指定した列 c に対して、「各要素を指定した桁数分だけ左シフトした結果を出力する」関数 shift_col() を作成しなさい。
この関数のプロトタイプ宣言は以下のようになる。

```
void shift_col(int c, int s);
  /* 引数 c の列に対する繰り返し処理を作る */
  /* 各要素の値を、引数 s 桁分だけシフトした結果を出力する */
```

main() での動作確認の例とその実行結果は以下のようになる。

```
[main() での処理]
  shift_col(2, 1);  /* 2 列目において各要素を 1 桁分左シフトした結果を出力する */
  shift_col(4, 2);  /* 4 列目において各要素を 2 桁分左シフトした結果を出力する */
[実行結果]
  6  12  16
  24  28  16
```

問題はここまで

（各 20 点）

定期試験の実施について

試験中に使用できるもの

- 筆記用具（メモ用紙は必要な人に配布）
- 演習室のコンピュータ一台（一つの机に一人の配置で、座る場所はどこでもよい）

試験中に参照できるもの

- 自分のホームディレクトリ（ホームフォルダ）以下に保存されているファイル

* **上記以外の情報を参照することはカンニング行為とする**
（例：USB で接続された機器に保存されているファイルの参照など）

* 試験中は、演習室外へのネットワークアクセスは遮断される

答案の提出

- 提出する全てのプログラムファイルの先頭行に、**自分の学科の出席番号と氏名**をコメントとして書く
- 保存したファイルは次のように「report」コマンドで提出する
（ちゃんと提出できた場合は、「Succeed.」と画面に表示される）

```
$ ~kogai/report 「提出先」 「プログラムファイル」
```

- 複数のファイルを提出する場合は、report コマンドを分けて提出する

例えば、`test1.c` と `test2.c` のファイルを提出したい場合は、次のように 2 回に分けて提出する

```
$ ~kogai/report kiso1term test1.c
$ ~kogai/report kiso1term test2.c
```

- 同じ問題に対して、複数の提出ファイルが存在した場合は、更新日時が新しい方を提出ファイルとする
- 提出するファイルは、誰から提出されたのか区別されるため、ファイル名は各自で自由に決めて良い
（ただし、提出するファイルの先頭には、出席番号と氏名を記入する）
- 「提出先」への提出は試験時のみ可能である

前期期末試験 模範解答（平均 82.1 点）

採点について コンパイル時にエラーとなる箇所は −4 点, 実行可能だか処理内容が問題の意図と違う箇所は −2 点を基本とする。

配点：1 ～ 5 各 20 点

ただし、4, 5 の問題において、「行→ row」「列→ column」の表記と実際の配列で一致していない箇所が数カ所あったため、“繰り返し処理の回数（2 点相当）” と “2 次元配列の添字（2 点相当）” に関わる部分は減点対象外とした。

```
#include <stdio.h>

#define ROW 5
#define COL 3
int numbers[COL][ROW] = {
    {6, 1, 3, 10, 6},
    {5, 9, 6,  1, 7},
    {6, 9, 8,  2, 4}
};

/* 関数のプロトタイプ宣言 */
void show_ascii(char ch, int n);
int to_int(int ch);
int count_row(int r, char cond, int x);
void shift_col(int c, int s);


/* [2] */
void show_ascii(char ch, int n)
{
    int i;
    for(i=0; i<=n; i++) {
        printf("%c(%x) ", (ch+i), (ch+i));
    }
    printf("\n");
}

/* [3] */
int to_int(int ch)
{
    int result;
    if(ch >= '0' && ch <= '9') {
        result = ch - '0';
    } else {
        result = -1;
    }
    return result;
}

/* [4] */
int count_row(int r, char cond, int x)
{
    int i, result;
    result = 0;
    for(i=0; i<ROW; i++) {
        if(cond=='e') {
            if(numbers[r][i]==x) { result ++; }
        } else if(cond=='l') {
            if(numbers[r][i]<x) { result ++; }
        } else if(cond=='g') {
            if(numbers[r][i]>x) { result ++; }
        }
    }
    return result;
}

/* [5] */
void shift_col(int c, int s)
{
    int i;
    for(i=0; i<COL; i++) {
        printf("%d  ", numbers[i][c] << s);
    }
    printf("\n");
}

int main(void)
{
    /* [1] */
    int num;
    num = 1;
    while(num != 0) {
        printf("16 進数 > ");
        scanf("%x", &num);
        if(num > 0x0 && num <= 0xffff) {
            printf("10 進数では %d\n", num);
        } else {
            printf("out of range\n");
        }
    }
    //show_ascii() の動作確認
    show_ascii('a', 3);
    show_ascii('Y', 10);
    //to_int() の動作確認
    printf("%d\n", to_int('0'));
    printf("%d\n", to_int('9'));
    printf("%d\n", to_int('/'));
    printf("%d\n", to_int(':'));
    //count_row() の動作確認
    printf("%d\n", count_row(0, 'e', 6));
    printf("%d\n", count_row(1, 'l', 7));
    printf("%d\n", count_row(2, 'g', 4));
    //shift_col() の動作確認
    shift_col(2, 1);
    shift_col(4, 2);

    return 0;
}
```