プログラミング基礎 前期中間試験

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 int 型の配列 test とマクロ SUM を以下のようにグローバル変数として宣言する。

```
#define N 5
#define SUM(x, y) (x+y)
int test[N] = {10, 25, 15, 5, 20};
```

この配列に対して、「全要素の値の合計を返す」関数 sum_array() を作成しなさい。ただし、マクロ SUM を使うこと。
この関数のプロトタイプ宣言は以下のようになる。

```
int sum_array();
  /* 繰り返し処理の中で、マクロ SUM を使って合計を求める */
  /* 求めた計算結果を戻り値とする */
```

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("sum_array: %d\n", sum_array());
  test[0] = 0;
  test[4] = 0;
  printf("sum_array: %d\n", sum_array());
[実行結果]
  sum_array: 75
  sum_array: 45
```

2 main() にて、scanf で入力した整数 a, b に対して「a^b」を出力するプログラムを作成しなさい。
実行結果は以下のようになる。

```
[実行結果 その 1]
  a > 3          (← 3 を入力した)
  b > 4          (← 4 を入力した)
  81.000000
[実行結果 その 2]
  a > 16         (← 16 を入力した)
  b > 2          (← 2 を入力した)
  256.000000
```

3 int 型の配列 binary とを以下のようにグローバル変数として宣言する。

```
#define N 5
int binary[N] = {0, 1, 1, 0, 1};   /* 2 進数 10110 を表現している */
```

この配列に対して、「配列 binary に格納されている値が表現している 2 進数を、10 進数に変換して、その値返す」関数 b_to_d() を作成しなさい。
ただし、配列 binary が表現している 2 進数は、0 番目が**最下位桁**となり、4 番目が**最上位桁**としており（つまり、通常の 2 進数表現とは桁が逆)、「binary[i] $\times 2^i$」（ただし、$0 \leq i \leq 4$）の合計で 10 進数の値が求まるとする。
この関数のプロトタイプ宣言は以下のようになる。

```
int b_to_d();
  /* 配列 binary の全要素に対する繰り返し処理を作る */
  /* べき乗の計算には関数 pow() 使って求める */
  /* 求めた計算結果を戻り値とする */
```

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("b_to_d: %d\n", b_to_d());
  binary[0] = 1;
  binary[3] = 1;
  printf("b_to_d: %d\n", b_to_d());
[実行結果]
  b_to_d: 22        (← 2 進数 10110 の 10 進数)
  b_to_d: 31        (← 2 進数 11111 の 10 進数)
```

4 char 型の配列 str1 を以下のようにグローバル変数として宣言する。

```
  char str1[50];
```

この配列に対して、「文字列に含まれているアルファベットの文字数を返す」関数 count_alpha() 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_alpha();
  /* 配列 str1 に格納されている文字列の繰り返し処理を作る */
  /* アルファベットかどうかの判別は、関数 isalpha() が使える */
  /* 求めた文字数を戻り値とする */
```

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("str1 > ");
  scanf("%s", str1);
  printf("count_alpha: %d\n", count_alpha());
[実行結果 その 1]
  str1 > June-03-2016        (←文字列を入力)
  count_alpha: 4
[実行結果 その 2]
  str1 > z=x+y
  count_alpha: 3
```

問題はここまで

1〜**4** 各 25 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 kiso1mid test1.c
$ ~kogai/report kiso1mid 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

前期中間試験 模範解答（平均 77.9 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 〜 4 各 25 点

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

#define N 5
#define SUM(x, y) (x+y)

int test[N] = {10, 25, 15, 5, 20};
int binary[N] = {0, 1, 1, 0, 1};
char str1[50];

int sum_array();
int b_to_d();
int count_alpha();

int sum_array()
{
    int i, result;
    result = 0;
    for(i=0; i<N; i++) {
        result = SUM(result, test[i]);
    }
    return result;
}

int b_to_d()
{
    int i, result;
    result = 0;
    for(i=0; i<N; i++) {
        result = result + binary[i] * pow(2, i);
    }
    return result;
}

int count_alpha()
{
    int i, result;
    result = 0;
    for(i=0; str1[i]!='\0'; i++) {
        if(isalpha(str1[i])) {
            result++;
        }
    }
    return result;
}

int main(void)
{
    int a, b;

    /* sum_array() の動作確認 */
    printf("sum_array: %d\n", sum_array());
    test[0] = 0;
    test[4] = 0;
    printf("sum_array: %d\n", sum_array());

    /* pow で x の y 乗 */
    printf("a > ");
    scanf("%d", &a);
    printf("b > ");
    scanf("%d", &b);
    printf("%f\n", pow(a, b));

    /* b_to_d() の動作確認 */
    printf("b_to_d: %d\n", b_to_d());
    binary[0] = 1;
    binary[3] = 1;
    printf("b_to_d: %d\n", b_to_d());

   /* count_alpha() の動作確認 */
    printf("str1 > ");
    scanf("%s", str1);
    printf("count_alpha: %d\n", count_alpha());

    return 0;
}
```