第12回
計算機基礎実習II

座席指定はありません。

計算機基礎実習II 2018 のウェブページから、以下の課題に自力で取り組んで下さい。

第11回の復習課題(rev11)

第12回の基本課題(base12)

第11回課題の解答例

ex11-3.c

関数 sqsum() は、2つの double 型の値a, b を引数として受け取ると、a^2 + b^2 の値を double型として返す関数

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

void pexit(void);
double sqsum(double, double);

int main(void) {
  double x, y;

  while (1) {
    printf("x y? ");
    scanf("%lf%lf", &x, &y);

    if (x==0 && y==0) pexit();

    printf("%f\n", sqrt(sqsum(x,y)));
  }
  return 1;
}
```

```
void pexit(void) {
  printf("終了します。\n");
  exit(0);
}

double sqsum(double x, double y) {
  return x*x + y*y;
}
```

ex11-4.c

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

int absum(int);

int main() {
  int a, b;

  do {
    printf("a b? ");
    scanf("%d%d", &a, &b);
  } while (abs(a) >= abs(b));

  printf("%d\n", absum(b) - absum(a));

  return 0;
}

int absum(int n) {..省略..}
```

ex11-5.c

関数 trcase() は、1つの char型の値を c を引数として受け取り、c が大文字アルファベット(A-Z)であれば小文字アルファベット(a-z)に変換し、小文字アルファベットであれば大文字アルファベットに変換して char型の返り値として返す関数

```
#include <stdio.h>

char trcase(char);

int main(void) {
  char c;

  while (1) {
    do {
      //printf("c? ");
      scanf("%c", &c);
    } while (c < 'A' || 'z' < c || ('Z' < c && c < 'a'));
    //      (c < 65  || 122 < c || (90  < c && c < 97 ));

    printf("%c => %c\n", c, trcase(c));
  }
  return 0;
}

char trcase(char c) {
  if ('A' <= c && c <= 'Z') return c+('a'-'A');
  else if ('Z' <= c && c <= 'z') return c-('a'-'A');
  else return c;
}
```

```
   :
 63  ?
 64  @
 65  A
 66  B
   :
 90  Z
 91  [
   :
 96  `
 97  a
 98  b
   :
122  z
123  {
   :
```

ex11-6.c

関数 max() は、2つの double型の値 a, b を引数として受け取り、そのうちの大きい方の値を返す double型の関数

関数 max4() は、4つの double型の値 a, b, c, d を引数として受け取り、そのうちのもっとも大きい値を返す double型の関数

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

double max(double, double);
double max4(double, double, double, double);

int main(void) {
  double a, b, c, d;

  while (1) {
    printf("a b c d? ");
    scanf("%lf%lf%lf%lf", &a, &b, &c, &d);

    printf("max: %lf\n", max4(a,b,c,d));

    if (a==b && c==d && a==c) return 0;
  }
  return 1;
}
```

```
double max(double a, double b) {
  if (a > b) return a;
  else return b;
}

double max4(double a, double b,
            double c, double d) {
  if (max(a,b) > max(c,d)) return max(a,b);
  else return max(c,d);
}
```

ex11-7.c

```
// n! を計算する再帰呼出し
#include <stdio.h>

int fact(int);

int main(void) {
  int n;

  printf("n? ");
  scanf("%d", &n);
  if (n < 0) return 1;

  printf("%d! = %d\n", n, fact(n));

  return 0;
}

int fact(int n) {
  if (n == 0)
    return 1;             // 0! = 1
  else
    return n*fact(n-1);   // n! = n * (n-1)!
}
```

第12回の内容

変数のスコープ

ローカル変数とグローバル変数

オンラインプログラミング環境の紹介（再）

変数のスコープ

変数のスコープ ＝ 変数の有効範囲

変数は「いつ生成されて」「いつ破棄されるか」
＝ プログラムのどこで参照・代入できるか

変数が宣言されてから、宣言されたブロック{ } が閉じるまで

```
#include <stdio.h>

int main(void) {
  printf("a=%d\n", a);

  int a = 1;

  printf("a=%d\n", a);
  return 0;
}
```

変数 a が宣言されたブロック

変数 a の宣言

変数 a のスコープ（有効範囲）

変数 a のスコープ外なのでエラー

スコープは変数ごとに決まる

変数が宣言されてから、宣言されたブロック{ } が閉じるまで

```
#include <stdio.h>

int main(void){
  int a = 1;
  printf("a=%d\n", a);

  if (a == 1) {
    int b = 10;
    printf("a=%d\n", ++a);
    printf("b=%d\n", b);
  }

  printf("a=%d\n", ++a);
  printf("b=%d\n", b);
  return 0;
}
```

変数 a の宣言

変数 b の宣言

変数 b のスコープ外なのでエラー

同じ名前で異なるスコープの変数

同名の変数もスコープが異なれば別の変数（関数もスコープを作る）

```
#include <stdio.h>

void inc(int);

int main(void){
  int a = 1;
  printf("a=%d\n", a);
  inc(a);
  printf("a=%d\n", a);
  return 0;
}

void inc(a) {
  a++;
}
```

main()関数の a のスコープ

a=1

a=1

inc()関数の a のスコープ

main() から inc(1)で呼ばれて a=2 となるがスコープが終わって捨てられる

同じ名前で重複するスコープの変数

同名の変数は内側スコープの変数として扱われる

```
#include <stdio.h>

int main(void){
  int a = 1;
  printf("a=%d\n", a);

  if (a == 1) {
    int a = 10;
    printf("a=%d\n", ++a);
    {
        int a = 100;
        printf("a=%d\n", a);
    }
    printf("a=%d\n", ++a);
  }

  printf("a=%d\n", ++a);
  return 0;
}
```

外層の変数 a のスコープ

中層の変数 a のスコープ

内層の変数 a のスコープ

ローカル変数とグローバル変数

ローカル変数 = ブロックによって制限された有効範囲をもつ
これまで扱ってきた変数はすべてローカル変数

グローバル変数 = すべてのスコープからアクセス可能
一見、便利そうだが、どこからでも変更できるのでバグの温床になりやすい（ご利用は計画的に）

```
#include <stdio.h>
void inc(void);
int a = 1;

int main(void) {
  printf("a=%d\n", a);
  a += 2;
  inc();
  printf("a=%d\n", a);
  return 0;
}

void inc(void) {
  a += 3;
}
```

グローバル変数 a の宣言

a=1

a=6

ex12-1.c

```
#include <stdio.h>

void fa(void);
int a = -1;

int main(void) {
  printf("a=%d\n", a);
  int a = 1;
  {
    int a = 10;
    printf("a=%d\n", a);
    fa();
    a = 20;
  }
  a = 2;
  printf("a=%d\n", a);
  return 0;
}

void fa(void) {
  {
    int a = 100;
    {
      int a = 200;
      printf("a=%d\n", a);
      a = 300;
    }
    printf("a=%d\n", a);
  }
  a = -2;
  printf("a=%d\n", a);
}
```

グローバル変数a=-1

main()関数の内側のa=10

fa()関数の内側のa=200

fa()関数の外側のa=100

グローバル変数a=-2

main()関数の外側のa=2

paiza.io

paiza.jp