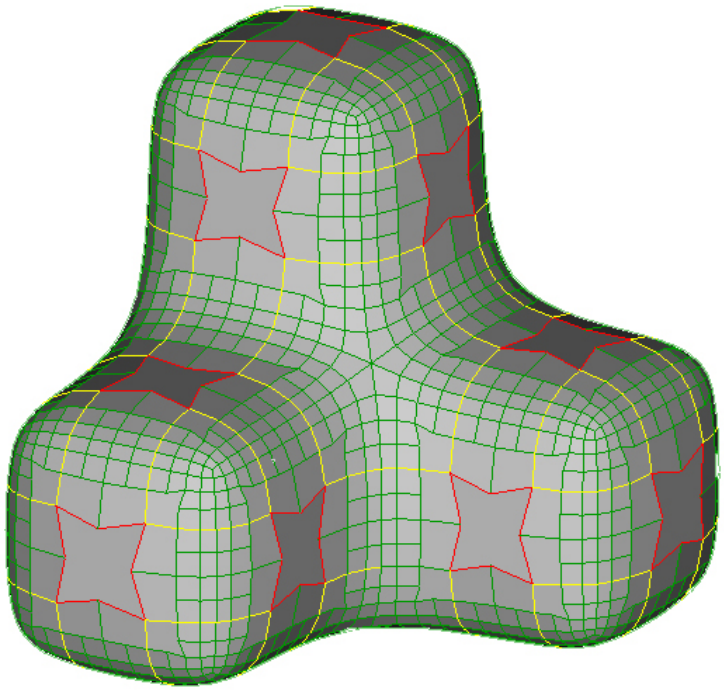
プログラミング

第4週



静岡大学
工学部機械工学科

知能・材料コース
ロボット・計測情報分野
臼杵 深

光電・精密コース
光ナノバイオ分野
居波 渉

講義の前に（重要な情報）

・講義の担当

第4週～第7週は臼杵が担当

・期末試験

クラスⅡ：2月9日（金）

・再試験

期末試験で60点未満の場合，再試験となる．

2月26日または2月27日の予定

講義アウトライン

- 復習
  - データ型(int, float, double) p. 37
  - 算術演算子(+, -, *, /, %) p. 56
  - `if`文 p. 76
  - 関係演算子 p. 57
- C言語の基礎
  - 特徴
  - ルール p. 34
  - 配列 p. 42

復習：for文による反復処理

◇ for文 教科書P-79

p. 79

- もっともよく用いられる制御文
- 指定回数の繰り返し
- 繰り返し処理をループ（loop：輪）

- for(初期化式； 継続条件式； 再設定式)

```
int i, s;
s=0;
for(i=1; i<=10; i++){
        s=s+i;
}
```

初期化

継続条件

インクリメント演算子
p. 59

- 初期化式 i=1
- 継続条件式 i<=10
- 再設定式 i++; /* i=i+1; と同じ意味 */

復習：for文の注意点

◯セミコロンを忘れない

```
for(i=1; i<=10; i++){
                s=s+i;
        }
```

◯{}を忘れない （カッコの数が合うように!!）

```
for(i=1; i<=10; i++){
                s=s+i;
        }
```

◯for()の行に ； を付けない

```
for(i=1; i<=10; i++);{
                s=s+i;
        }
```

課題2−2

```
#include <stdio.h>
int main(void)
{
    int i,s;
    s=10;
    for(i=3; i<10; i++){
        s=s+i;
        printf("i=%d, s=%d¥n",i,s);
    }
    printf("i=%d, s=%d¥n",i,s);
    return 0;
}
```

課題2−2

```
#include <stdio.h>
int main(void)
{
    int i,s;
    s=10;
    for(i=3; i<10; i++){
        s=s+i;
        printf("i=%d, s=%d¥n",i,s);
    }
    printf("i=%d, s=%d¥n",i,s);
    return 0;
}
```

i=3, s=13
i=4, s=17
i=5, s=22
i=6, s=28
i=7, s=35
i=8, s=43
i=9, s=52
i=10, s=52

復習：変数 - その2（実数） p-37

【説明】 p-37

データ（数値）を記憶しておく箱を変数と言う。変数は使用する前に保持するデータの型を指定して宣言しなければならない。すでに整数を保存するデータ型として int を学んだが、実数(小数)を保存するデータ型として

float、double

というデータ型がある。C言語では小数のデータ型のことを「浮動小数点型」と呼んでいる。floatとdoubleの違いは小数を保存するサイズでdoubleの方がより多くの桁数を保持できる。絶対的なサイズ（何桁までデータを保持するか）は処理系に依存する。

【用例】

```
float x;  /* float型の変数 x を宣言する。*/
double y; /* double型の変数 x を宣言する。*/
x = 0.05f; /* 変数に値を代入する。          */
y = 3.5;   /* 変数に値を代入する。          */
```

接尾語
p. 29

復習：変数

◇ 変数と代入演算子

【説明】

変数の型　変数名;

```
ex) int i;
```

データ型	バイト幅	表現範囲
char	1	-128～127
int	4	-2,147,483,648～2,147,483,647
float double	4 8	10の-37乗から10の38乗（有効6桁） 10の-307乗から10の308乗（有効15桁）

整数値のオーバーフロー、丸め誤差に注意 p. 41

復習:算術演算子 – その1(+ - * /)

【説明】p-56

演算子	説明	例
+	正符号	a = +b;
-	負符号	a = -b;
+	加算(足し算)	a = b + c;
-	減算(引き算)	a = b – c;
*	乗算(かけ算)	a = b * c;
/	除算(割り算)	a = b / c;
%	余り	a = b % c;

p. 56

算術演算子は数値を計算する。結果として数値を得る。整数除算を行うと余りは切り捨てられる。また整数に対して % 演算子を使うと余りを求めることができる。0で除算すると結果は不定になる。

復習：printf()の変換文字列

教科書 p-112

%d　10進数で出力する：int

%f　浮動小数点数で出力する：float, double

接尾語 p. 40

【用例】

```
int n = 10;        /* int型の変数 n の宣言                    */
printf("%d",n); /* 10進数として出力する。 */
```

```
float f = 10.0f; /* float型の変数 f の宣言                  */
printf("%f",f); /* 浮動小数点数で出力する。                  */
```

```
double d = 10.0;  /* double型の変数 d の宣言                 */
printf("%f",d);   /* 浮動小数点数で出力する。                  */
```

p.117, "%lf" でもOK

復習：printf()の注意事項

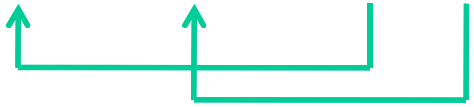
◯ 複数の変数を出力する: a, bの2つの値をひとつの文で出力する

```
int a,b;
a=2;
b=3;
printf(“a=%d, b=%d¥n”, a, b);
```

復習：printf()の注意事項

◯ 複数の変数を出力する: a, bの2つの値をひとつの文で出力する

```
int a,b;
a=2;
b=3;
printf(“a=%d, b=%d¥n”, a, b);
```



出力

a=2, b=3

```
printf(“a=%d, b=%d, c=%f ¥n”, a, b, c);
```

復習：scanf()

教科書 p-117

p. 117

◇ scanf()関数

【説明】

キーボードからの入力を受け付け変数にその値をセットする。プログラムの実行がこの行に達するとプログラムは一端そこで停止し、キーボードからの入力待ち状態になる。

【用例】

```
int n;              /* 整数型の変数 n の宣言                    */
scanf("%d",&n); /* キーボードから入力された整数を n という */
                    /* 変数にセットする。                       */
```

※ 演習では scanf_s() を使用する

復習：scanf()の変換文字列

教科書 p-117

%d　10進数を読む: int型変数を読む

%f　float型変数に実数を読む

%lf double型変数に実数を読む

p. 117

【用例】

```
int n;              /* int型の変数 n の宣言                    */
scanf("%d",&n); /* キーボードから入力された整数を n という */
                    /* 変数にセットする。                      */
```

```
float f;            /* float型の変数 f の宣言                  */
scanf(“%f”,&f); /* キーボードから入力された実数を f という */
                    /* 変数にセットする。                      */
```

```
double d;           /* double型の変数 d の宣言                 */
scanf("%lf",&d); /* キーボードから入力された整数を d という */
                    /* 変数にセットする。                      */
```

復習：scanf()の用例

```
リスト 教科書 p-117

#include<stdio.h>

int main(void)

{

        int n;

        scanf(“%d”,&n);

        printf(“%d¥n“,n);

}
```

p. 117

10進数で読み込む

10進数で出力

復習：scanf()の注意点

○ &を忘れない。

```
int n;
scanf(“%d”,&n);
printf(“%d¥n“,n);
```

○ ¥nや文字を付けない。

×
```
scanf(“n=%d¥n”,&n);
 printf(“n=%d¥n“,n);
```

○ 二つの変数を同時に読み込む場合のやり方

```
int a, b;
scanf(“%d%d”,&a,&b);
```

×
```
scanf(“a=%d, b=%d”,&a,&b);
printf(“a=%d, b=%d”,a,b);
```

復習：printf(), scanf()の変換仕様

	printf( )	scanf( )
int	%d	%d
float	%f	%f
double	%f (%lfでもOK)	%lf

※double型の場合に注意。正しくは %f だが
%lf でもエラーは生じない。

復習:if文:書式1

◇ if文　教科書 p-76

【説明】 if文の基本的な動作は、条件にしたがって処理を２方向に分岐する。省略や多重構造にすることによって３つのバリエーションがある。

書式１

```
if (条件) {
        文
    }
```

もし条件が真(0以外)なら文を実行する 。偽(0)ならなにもしない。

文が複数のときは {} を用いる。

【用例】

```
int i;
 for(i=0; i<5; ++i){
     if(i == 0){
         printf("a");
     }
 }
```

復習:if文:書式2

◇ if文

書式２

if(条件){

文１}

else{

文２}

もし条件が真(0以外)なら文１を実行する。偽(0)なら文２を実行する。

【用例】

```
int i;
for(i=0; i<5; ++i)
    if(i == 0){
        printf("a");
    }
    else{
        printf("b");
    }
```

復習:if文:書式3

◇ if文

書式３

```
if(条件１){       /* もし条件１が真(0以外)なら文１を実行する。 */
        文１}
else if(条件２){ /* そうではなく条件２が真(0以外)なら文２を実行する。 */
        文２}
          ・
          ・
else if(条件n){ /* そうではなく条件nが真(0以外)なら文nを実行する。 */
        文n }
  else{
        文 }  /* そうでなければ(0)文を実行する。 */
```

復習:if文:書式3 の用例

◇ if文

書式３

【用例】

```
int i;
for(i=0; i<5; ++i){
    if(i == 0){
        printf("a");
    }
    else if(i != 4){
        printf("b");
    }
    else{
        printf("c");
    }
}
```

実行結果　?

復習:if文:書式3 の用例

◇ if文

書式３

【用例】

```
int i;
for(i=0; i<5; ++i){
    if(i == 0){
        printf("a");
    }
    else if(i != 4){
        printf("b");
    }
    else{
        printf("c");
    }
}
```

実行結果　**abbbc**

i	満たす条件	出力
0	i == 0	a
1	i != 4	b
2	i != 4	b
3	i != 4	b
4	else	c

・else文は省略できます。

・elseに条件はつけることはできない。

復習:関係演算子

◇ 関係演算子 p.57

p. 57

【説明】

関係演算子は、２つのオペランド（演算の対象）の大小関係（どちらの数が多いか）や等値関係（２つの数値が等しいか）を判定して、真偽値（式が成り立つかどうか）を得る。２つの数の関係が正しければ１、正しくなければ０になる。

演算子	説明	例
<	小さい	`if(a < b)`
<=	小さいか等しい	`if(a <= b)`
>	大きい	`if(a > b)`
>=	大きいか等しい	`if(a >= b)`
==	等しい	`if(a == b)`
!=	等しくない	`if(a != b)`

C言語の特徴 p-2

第1章

1. 小文字でプログラムを書く
2. 簡潔な表現ができる （i++, a=b=c=100 など）
3. 演算子が豊富（算術演算子、インクリメント演算子）
4. ポインタを用いる
5. データ型が豊富（int, float, double …）
6. 関数で構成される
7. 構造化制御文が備わっている
8. プリプロセッサ付きである
9. 入出力処理や文字列処理は関数で行う
10. 特殊文字の表現が可能
11. 関数プロトタイプを宣言する

C言語のルール p-34

1. プログラムは関数で構成される
2. プログラムはmain関数が１つ必要である
3. 関数は　{　で始まり　}　で終わる
4. 文には；(セミコロン)が必要である
5. コメントは /* と */ で囲む
6. 変数は宣言してから使用する
7. 変数の値を出力するには書式指定が必要である
   例) %d, %f, %lf
8. プログラムは字下げをすると見やすくなる
9. 名前は英数字と_(下線)を使って書く
12. 不可視コードはエスケープシーケンスで記述できる
   ¥n （改行）, ¥t（タブ）など。

p. 34

配列 – その1（1次元配列）

【説明】 p.42

p. 42

配列は、同じ名前で操作される同じ型のデータを集めたもの．1次元配列の宣言の基本的な書式は次のとおり．

データ型名　変数名[サイズ];

配列確保の例

```
int a[10];          /* int型変数10個からなる配列a      */
int b[10], c[5]; /* int型変数10個からなる配列bと    */
                    /* 5個からなる配列 c                 */
double d[20];     /* double型変数20個からなる配列 d */
```

配列 – その1（1次元配列）

このとき確保される要素は0からはじまる。したがって「int a[10];」と宣言したとき利用できる配列は、a[0]～a[9]までの10個である。配列の使い方は普通の変数と同様で、a[5]とすればaという配列の6番目の要素に代入したり値を参照したりできる。 5のことを添字という.

またコンパイラは、実行時に配列の境界をチェックしない。それは、「int a[10];」の宣言で使える配列要素はa[0]～a[9]までだが、このときに、

```
a[10] = 100;
a[25] = 100;
```

といった代入をしても、強引に実行する。それはa[0]から11番目や26番目の要素に該当するアドレスの内容を無理矢理書き変えることになる。
したがって、プログラムを書く際は境界を越えた代入をしないように注意しなければならない。

配列:用例 その1

【用例】リスト p-43

p. 43

```
#include <stdio.h>
int main(void)
{
    int n; int dt[10];
    dt[0]=10; dt[9]=99; n=5; dt[n]=55; dt[2]=dt[9];

    printf("dt[0]=%d¥n",dt[0]);
    printf("dt[2]=%d¥n",dt[2]);
    printf("dt[5]=%d¥n",dt[5]);
    printf("dt[9]=%d¥n",dt[9]);
    return 0;
}
```

配列:用例 その2

配列の値の加算

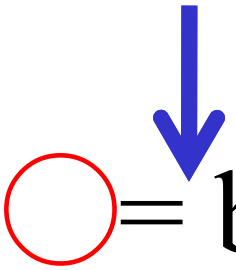
```
#include <stdio.h>
int main(void)
{
    int i,s;
    int a[10]={1,2,3,5,7,11,13,17,19,23};
    s=0;
    for(i=0; i<=9; i++)
        s+=a[i]; /* s=s+a[i]; と同じ  */

    printf("sum=%d¥n",s);
    return 0;
}
```

配列の初期化 p. 51

複合代入演算子 p. 61

複合代入演算子 (p. 61)

a = a ○ b （○は演算子）



例

```
mylongdata*= 20;
mylongdata=mylongdata*20;
```

Sizeof 演算子 (p. 67)

p. 67

```
char  cdt, ss[10];
int   n, idt, ii[10];
double ddt;

n = sizeof cdt;   → n=1
n = sizeof idt;   → n=4
n = sizeof ss;    → n=10 (1byte X 10)
n = sizeof ii;    → n=40 (4byte x 10)
n = sizeof ddt;   → n=8
n = sizeof(char); → n=1
n = sizeof(int);  → n=4
n = sizeof(double);  → n=8
n = sizeof “abcde”;  → n=6（ナル文字¥0を含む）
```

データ型のバイト数

p. 67

バイトとは何か述べよ。また、教科書p-67のsizeof関数を利用して，int型，float型，double型のバイト数を表示するプログラムを作成せよ。

```
#include <stdio.h>
int main(void)
{
    printf("%d¥n",sizeof(int));
    printf("%d¥n",sizeof(float));
    printf("%d¥n",sizeof(double));
    return 0;
}
```

まとめ

- C言語の基礎
  - 特徴
  - ルール
  - 配列

配列：課題４

課題4の2.4

教科書P-51の配列の初期化を利用して以下の数値の2乗を計算して出力するプログラムを作成せよ。

配列の初期化の例

```
int a[10]={1,2,3,5,7,11,13,17,19,23};
```

配列：課題4の解答

```
#include <stdio.h>
int main(void){
    int i;
    int a[10]={1,2,3,5,7,11,13,17,19,23};
    int b[10];

    for(i=0; i<=9; i++){
        b[i]=a[i]*a[i];
    }
    for(i=0; i<=9; i++){
        printf("%d¥n",b[i]);
    }
    return 0;
}
```

配列の初期化

２乗を計算してb[i]に代入

b[i]の値を出力