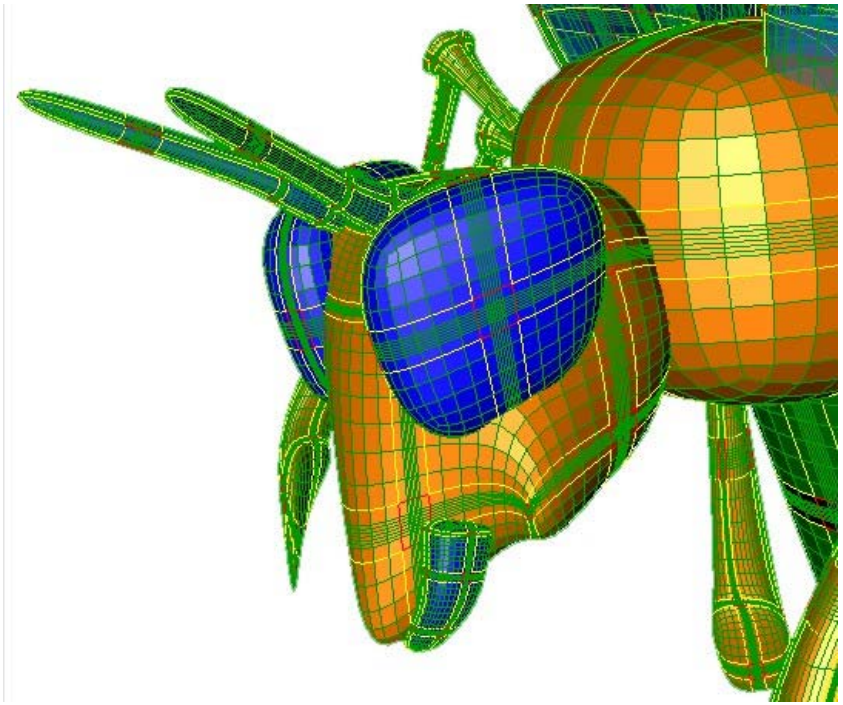
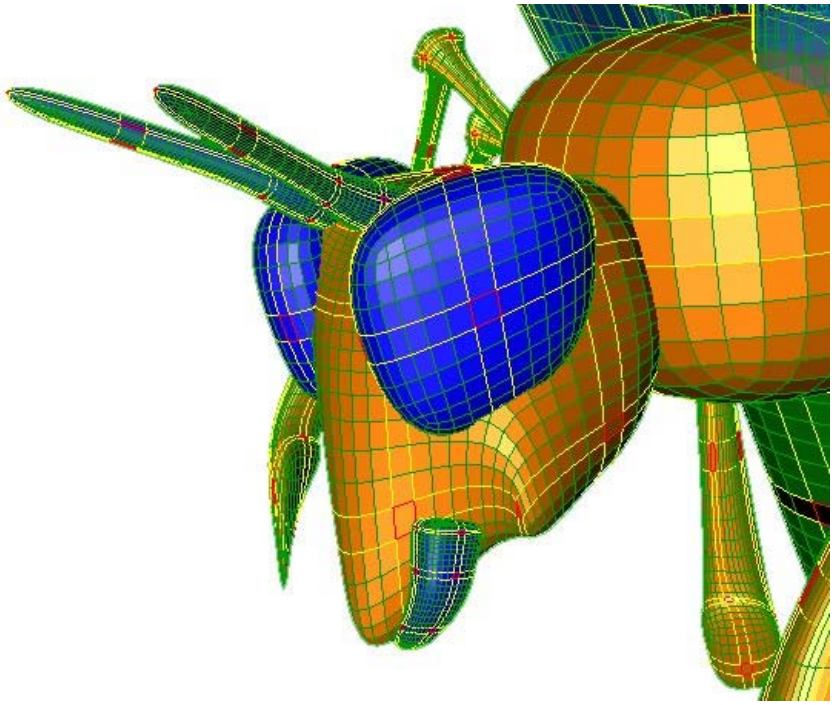
プログラミング

第7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

期末試験

- 日程：2018年2月9日（金）10:20-11:50
- 場所：1-31教室
- 時間：90分
- ※持ち込み不可
- ※遅刻厳禁
- ※学生証持参すること
- ※紛らわしい行為は不正とみなす
- ※60点未満の場合，再試験（2月26日または2月27日の予定）

講義アウトライン

- 復習
  - switch文 (p. 87) , while文(p. 82) , do-while文(p. 84)
  - break文, continue文
- C言語の基礎
  - キャスト演算子（p.62)
  - break文, continue文 (p. 92)
  - 1文字入出力　getchar(), putchar()　(p. 97)
  - EOF　(End of File) (p. 99)
  - NULL (p. 104)
  - 1行入出力 gets(), puts() (p. 102)

復習：switch文による多方向分岐

```
switch(式) {
  case 定数式1:
      文1 break;
  case 定数式2:
      文2 break;
      . . .
  default:
          文
}
```

復習：switch文による多方向分岐

- 式の値を評価し，その値と一致する定数式のcase部にジャンプ、該当する文を実行
- break文に出会うと処理を終了してswitch文全体を終了
- 一致する定数式がないときはdefault部に記述された文を実行
- default部は不要なら省略可能

```
i=j%2;
switch(i){
  case 0:
    printf(“偶数¥n”);
    break;
  case 1:
    printf(“奇数¥n”);
}
```

復習：switch文の例

```
#include <stdio.h> int main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf(“----%d¥n”,a);
  switch(a){
    case 1: printf(“a=1¥n”); break;
      . . .
    default: printf(“others¥n”);
    }
  }
  return 0;
}
```

復習：switch文の例

```
#include <stdio.h> int main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf("----%d¥n",a);
  switch(a){
    case 1: printf("a=1¥n"); break;
      . . .
    default: printf("others¥n");
    }
  }
  return 0;
}
```

実行結果

```
----1
a=1
----2
others
----3
others
----4
others
----5
others
```

復習：switch文の定数式

switch(式)　　式は整数を結果とする

```
int a;
switch(a) . . . 正しい
switch(a+10)    正しい

float f;
switch(f) . . . 誤り
```

case句　整数，文字定数，定数の式

正

```
case 5:   case ‘A’:  case ‘B+10’:
```

誤

```
case 5.6:  case a>10:  case “ABC”:
```

復習：switch文例

```
#include <stdio.h> int main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf(“----%d¥n”,a);
  switch(a){
    case 1: printf(“a=1¥n”); break;
    case 3: printf(“a=3¥n”); break;
    default: printf(“others¥n”);
    }
  }
  return 0;
}
```

復習：switch文例

```
#include <stdio.h> int
main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf(“----%d¥n”,a);
  switch(a){
    case 1: printf(“a=1¥n”);
    case 3: printf(“a=3¥n”);
    default: printf(“others¥n”);
    }
  }
  return 0;
}
```

実行結果

```
----1
a=1
----2
others
----3
a=3
----4
others
----5
others
```

復習：switch文例

```
int main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf(“----%d¥n”,a);
  switch(a){
    case 1:
    case 2:
      printf(“a=1 or 2¥n”);
      break;
    default: printf(“others¥n”);
    }
  }
  return 0;
}
```

復習：switch文例

```
int main(void)
{
  int a;
  for(a=1; a<=5; a++){
  printf(“----%d¥n”,a);
  switch(a){
    case 1:
    case 2:
      printf(“a=1 or 2¥n”);
      break;
    default: printf(“others¥n”);
    }
  }
  return 0;
}
```

実行結果

```
----1
a=1 or 2
----2
a=1 or 2
----3
others
----4
others
----5
others
```

復習：while文の使用例

```
#include <stdio.h>
int main(void)
{
    int i, dt[]={100,200,300,-1,400};
    i=0;
    while(dt[i]!=-1){
        printf(“%d¥n”,dt[i]);
        i++;
    }
    return 0;
}
```

ループに入る条件を
前もって判定

復習：while文の使用例

```
#include <stdio.h>
int main(void)
{
    int i, dt[]={100,200,300,-1,400};
    i=0;
    while(dt[i]!=-1){
        printf(“%d¥n”,dt[i]);
        i++;
    }
    return 0;
}
```

実行結果：
100
200
300

復習：do-while文の使用例

```
#include <stdio.h>
int main(void)
{
    int n; char ss[]="ABCDE";
    printf("%s¥n",ss);
    n=-1;
    do{                         /* char 文字型 */
        ++n;
        printf("%d¥n", ss[n]);
    }while(ss[n]!='¥0');
    return 0;
}
```

文字列は最後に¥0を持つ p.48

復習：do-while文の使用例

```
#include <stdio.h>
int main(void)
{
    int n; char ss[]="ABCDE";
    printf("%s¥n",ss);
    n=-1;
    do{                    /* char 文字型 */
        ++n;
        printf("%d¥n", ss[n]);
    }while(ss[n]!='¥0');
    return 0;
}
```

実行結果

```
ABCDE
65
66
67
68
69
0
```

復習:キャスト演算子　(p. 62)

（型）式　「式」で表すデータが指定の「型」に変換される

（例1）

```
int idt;
double ddt = 123.456;

idt = (int) ddt;

/* idtには整数123が格納される */
```

復習：キャスト演算子　(p. 62)

（型）式　「式」で表すデータが指定の「型」に変換される

（例２）

```
int a=5, b=3;
double ddt1, ddt2, ddt3;

ddt1 = a/b;
ddt2 = (double)a/b;
ddt3 = (double)(a/b);
```

それぞれ何が格納される？

復習:キャスト演算子　(p. 62)

（例２）

```
int a=5, b=3;
double ddt1, ddt2, ddt3;

ddt1 = a/b;
ddt2 = (double)a/b;
ddt3 = (double)(a/b);
------------------------------
/* ddt1には実数1.000000が格納される　商に丸められる*/
/* ddt2には実数1.666666が格納される　除算より先にキャスト*/
/* ddt3には実数1.000000が格納される　キャストより先に除算*/
```

break文　p.92

break;

- **switch**文、または　**for**文，**while**文，**do-while**文などのループから脱出
- **switch**文以外で用いる**break**文は通常**if**文と併用
- ループがネスト(多重ループ)構造になっているときは、**break**文の存在するループを1つだけ抜ける

```
i=10;
while(1){
   printf(“%d¥n”,i);
   if(i==13)break;
   ++i;
}
```

break文　p.92

break;

- **switch**文、または　**for**文，**while**文，**do-while**文などのループから脱出
- **switch**文以外で用いる**break**文は通常**if**文と併用
- ループがネスト(多重ループ)構造になっているときは、**break**文の存在するループを1つだけ抜ける

```
i=10;
while(1){
  printf(“%d¥n”,i);
  if(i==13)break;
  ++i;
}
```

実行結果
10
11
12
13

break文例

```
#include <stdio.h> int
main(void)
{
  int i,j;
  for(i=1; i<=4; i++){
    printf(“i=%d: ”,i);
    for(j=1;j<=4;j++){
      printf(“j=%d “,j);
      if(j==2) break;
    }
    printf(“¥n”);
  }
  return 0;
}
```

break文例

```
#include <stdio.h> int
main(void)
{
  int i,j;
  for(i=1; i<=4; i++){
    printf(“i=%d: ”,i);
    for(j=1;j<=4;j++){
      printf(“j=%d “,j);
      if(j==2) break;
    }
    printf(“¥n”);
  }
  return 0;
}
```

実行結果

```
i=1:  j=1  j=2
i=2:  j=1  j=2
i=3:  j=1  j=2
i=4:  j=1  j=2
```

continue文　p.92

continue;

- **for**文、または **while**文, **do-while**文　のなかで使用
- ループ処理をその回だけスキップし、以降の処理は継続
- **continue**記述位置から、ループ終端部までスキップ

```
sum=0;
for(i=1;i<=5;i++){
  printf("continue i=%d¥n",i);
  if(i==3)continue;
  sum=sum+i; /* sum+=i; */
  printf("     sum=%d¥n");
}
```

continue文　p.92

continue;

- **for**文、または **while**文, **do-while**文　のなかで使用
- ループ処理をその回だけスキップし、以降の処理は継続
- **continue**記述位置から、ループ終端部までスキップ

```
sum=0;
for(i=1;i<=5;i++){
  printf("continue i=%d¥n",i);
  if(i==3)continue;
  sum=sum+i; /* sum+=i; */
  printf("     sum=%d¥n");
}
```

実行結果

```
continue i=1
    sum=1
continue i=2
    sum=3
continue i=3
continue i=4
    sum=7
continue i=5
    sum=12
```

continue文例

```
#include <stdio.h> int
main(void)
{
  int i,j;
  for(i=1; i<=4; i++){
    printf(“i=%d: ”,i);
    for(j=1;j<=4;j++){
      printf(“j=%d start¥n“,j);
      if(j==2) continue;
      printf(“j=%d end¥n“,j);
    }
    printf(“¥n”);
  }
  return 0;
}
```

continue文例

```
#include <stdio.h>
int main(void)
{
  int i,j;
  for(i=1; i<=4; i++){
    printf("i=%d: ",i);
    for(j=1;j<=4;j++){
      printf("j=%d start¥n",j);
      if(j==2) continue;
      printf("j=%d end¥n",j);
    }
    printf("¥n");
  }
  return 0;
}
```

実行結果

```
i=1 j=1 start
j=1 end
j=2 start
j=3 start
j=3  end
j=4 start
j=4 end

i=2 j=1 start
j=1 end
j=2 start
j=3 start
j=3  end
j=4 start
j=4 end
```

実行結果続き

```
i=3 j=1 start
j=1 end
j=2 start
j=3 start
j=3  end
j=4 start
j=4 end

i=4 j=1 start
j=1 end
j=2 start
j=3 start
j=3  end
j=4 start
j=4 end
```

1文字入出力 p.97

```
#include <stdio.h>


int main(void)
{
  int ch;
  ch=getchar();
  putchar(ch);
  return 0;
}
```

注意　**getchar**入力は必ず**int**型で受ける

EOF　p.99

```
#include <stdio.h>

#include <ctype.h>  /* for toupper() */
int main(void)
{
  int ch;
  ch=getchar();
  while(ch!=EOF){
    ch=toupper(ch);
    putchar(ch);
    ch=getchar();
  }
  return 0;
}
```

EOF: End Of File (p.99)

getchar()関数で入力終了を知るときはEOFを使う

EOF　p.99

実行結果（なぜこうなる？１文字入出力なのに1行ずつ処理されている！）

```
（入力）FORTRAN and Pascal
（出力）FORTRAN AND PASCAL
（入力）Prolog or LISP
（出力）PROLOG OR LISP

^Z
^Z Ctrl+Z
```

getchar()関数で入力終了を知るときはEOFを使う

Ctrl+Zを入力

EOF End Of File: ファイルが終わりになったら返される値

```
#define EOF     (-1)
```

バッファリング処理　p.100

バッファリング処理

1文字読み込み指示のときも、1行単位でまとめて読み込み、
その後で内部的に1文字ずつ処理する。

While + getchar　p.101

前ページのプログラムを以下のように変える　？

```
#include <stdio.h>
#include <ctype.h>
main()
{
  int ch;

  while((ch=getchar())!=EOF){
    ch=toupper(ch);
    putchar(ch);
  }
}
```

←この記述が定石
連続文字入力の
基本パターン
ch=getchar()の記述
が1回でOK

1行入出力 p.102

```
#include <stdio.h>

int main(void)
{
  char ss[80];
  gets(ss);
  puts(ss);
  return 0;
}
```

注意　**enter**キーが押されるまで文字列を格納する

puts は最後に改行を含む，ss[]は十分な長さを確保する

NULL p.104

```
#include <stdio.h>

int main(void)
{
  char ss[80];
  while(gets(ss) != NULL){
     puts(ss);
    puts(ss);
  }
  return 0;
}
```

gets()関数では入力終了になるとNULLが返される
Ctrl+Z

注意 CNTL Z で NULL を返す

復習:scanf()の変換文字列

教科書 p-116

%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

【用例】

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

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

復習：printf()の変換文字列

教科書 **p-112**

%d 10進数で出力する： **int**

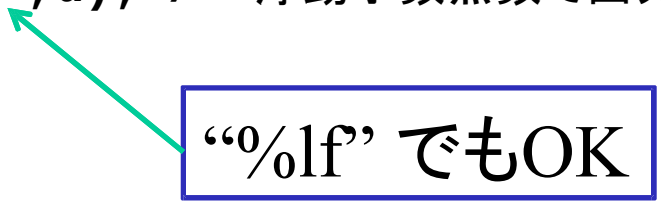
%f 浮動小数点数で出力する： **float, double**

【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。                    */
```



printf()のオプション指定 p.114

- フィールド指定子
  - %d　　10進数で出力
  - %8d　　8桁の10進数で出力（数が7桁以下の場合右寄せ）
  - %-8d　　左寄せ8文字幅
- 精度指定子
  - %f　　浮動小数点数を標準の桁数で表示
  - %10.2f　全体を10桁、小数点以下2桁
  - 例　　%9.2f　　987654.32

#define p.186

#defineとは？

【説明】 **define**は文字列の置き換えを行う。次のような書式になる。

```
#define 文字列1 文字列2 たとえ
```

ば、次のように定義する。

```
#define SIZE 500
```

こうするとプログラム中の“**SIZE**”という文字はコンパイルの時に“**500**”という数値 に置き換わる。たとえば以下のように用いる。

```
#define SIZE 500
int a;
a = SIZE + 50;
```

こうすると、**a**には**550**が入る。一般に**define**で定義される名前(上の例では**SIZE**)は普通の変数と区別するために大文字で書くのが慣例である。

まとめ

- C言語の基礎
  - コンソール入出力
  - getchar(), putchar()
  - gets(), puts()

小テスト

次のプログラムを作成せよ。

1. キーボードから入力した文字列を2回出力して、Ctrl+Zが入力されたら処理終了する。gets(), puts()を使用すること。

2. 入力した実数値を小数点も含めて全体の桁数12, 小数点以下2桁で出力する。