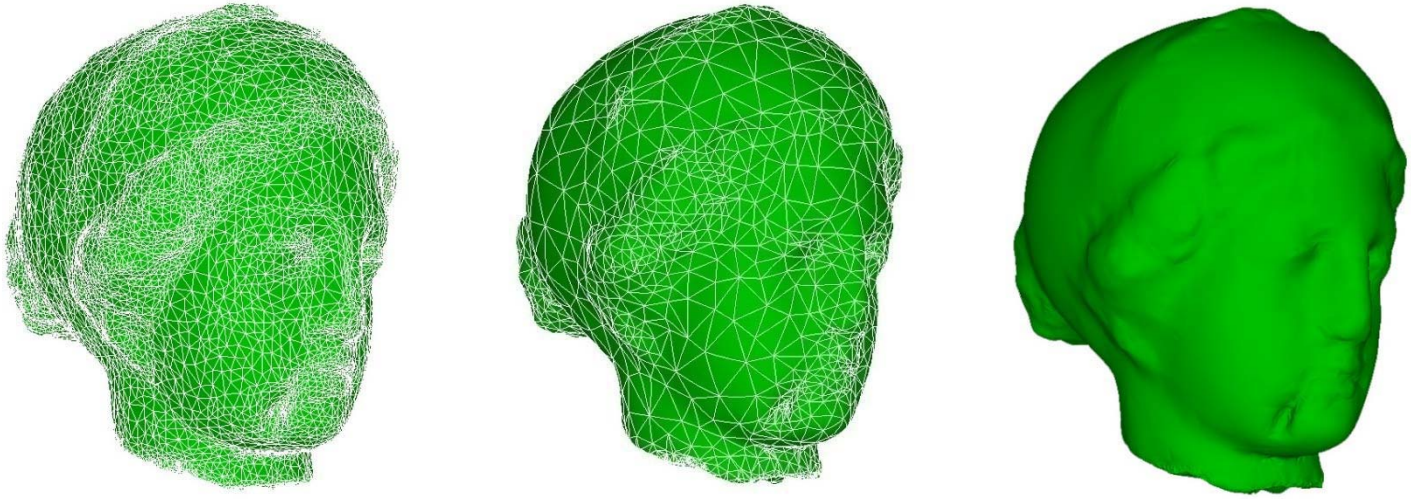
プログラミング

第6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
  - 配列の誤用
  - 論理演算子　(p. 58)
  - インクリメント/デクリメント演算子 (p. 59)
  - 演算子の優先順位と結合規則 (p. 68)
- C言語の基礎
  - for文による無限ループ (p. 81)
  - while文 (p. 82)
  - do-while文 (p. 84)
  - switch文 (p. 87)
  - キャスト演算子（p.62)

復習:配列の誤用 その1

以下から間違いを指摘せよ!

```
#include <stdio.h>
int main(void)
{
    int i;
    int a[];
    float s;
    float b[]={1.2f,2.3f,3.5f};

    double c[2]={2.5,3.4,1.3};
    b[]=6.7f;
    b[-1]=2.6f;
    b[3]=5.8f;
```

復習:配列の誤用 その1

```
#include <stdio.h>
int main(void)
{
    int i;
    int a[];/* 誤り:初期化されない配列は必ずサイズを宣言する.     */
    float s;
    float b[]={1.2f,2.3f,3.5f};  /* 正しい宣言 */
                                 /* 配列のサイズは3となる */
    double c[2]={2.5,3.4,1.3};/* 誤り:サイズが初期化と異なる    */
    b[]=6.7f;          /* 誤り:添字がないのでどの値か決められない  */
    b[-1]=2.6f;        /* 誤り:負の添字は許されない  */
    b[3]=5.8f;         /* 誤り:サイズは3なので, 添字は0,1,2のみ  */
```

復習:配列の誤用 その2

以下から間違いを指摘せよ!

```
b[0]=b[1]+b[];

/* 配列に格納された値の平均を求める */
s=0;
for(i=1;i<=3;++i)
{
    s+=b[i];
}
s/=3;
printf("平均 = %f￥n",s);
return 0;
}
```

復習：配列の誤用 その2

```
    b[0]=b[1]+b[];  /* b[]は誤り：添字がない */

    /* 配列に格納された値の平均を求める */
    s=0;
    for(i=1;i<=3;++i)  /* 誤り：正しくは for(i=0;i<3;++i) */
    {
        s+=b[i]; /* s=s+b[i]; と同じ */
    }
    s/=3; /* 正しい：s=s/3; と同じ */
    printf("平均 = %f￥n",s);
    return 0;
}
```

複合代入演算子 p. 61

追加:配列の誤用 その3

```
#include <stdio.h>
int main(void)
{
        int i=5;
        int a[10],b[i];
        a[i]=i;
        ...
}
```

追加:配列の誤用 その3

```
#include <stdio.h>
int main(void)
{
        int i=5;
        int a[10],b[i];  /* 誤り 配列宣言の際のサイズ
                            指定に変数は使えない   */
        a[i]=i;
        ...
}
```

復習:論理演算子　p-58

論理演算子

p. 58

真偽値を否定したり、複数の条件を組み合わせる。

```
if(a == 10 && b == 20){ /*  aが10でかつbが20ならば  */
  ...
}
```

p. 96　if(flg==0)と同じ

演算子	説明	使用例
!	否定	`if(!flg) /* flgが0であれば真 */`
&&	論理積	`if(10 < a && a < 20)`
||	論理和	`if(a == 2 || a == 4)`

復習：論理演算子の使用例

```
#include <stdio.h>
int main(void) {
  int a;
  for(a = 1; a <= 5; a++) {
    printf("----%d¥n", a);
    if(2 <= a && a <= 4){
        printf("2以上かつ４以下¥n");
    }
    if(a < 2 || 4 < a){ printf("2未満
        または４より大¥n");
    }
  }
  return 0;
}
```

復習：インクリメント/デクリメント演算子

演算子	説明	使用例
`++`	1 加算	`++a;` あるいは `a++;`
`--`	1 減算	`--a;` あるいは `a--;`

`a++;` あるいは `++a;` は `a=a+1;` と等価
`a--;` あるいは `--a;` は `a=a-1;` と等価

例

```
a=++b;   -> b=b+1; a=b    （前置型）

a=b++;   -> a=b; b=b+1;    （後置型）
```

復習：算術演算子
－その２（優先順位と結合規則）

【説明】これまでに説明した算術演算子（他にも演算子はあるが）には、優先順位が ある。これはたとえば、「1+2*3」では

2*3が先に計算されて、結果は7になる といったことである。もし「1+2」

を優先して演算したいときは、優先順位が

もっとも高い （ ） を用いて、順位を切り替える。すなわち「(1+2)*3」とする。

復習:結合規則

もうひとつ演算子には結合規則というものがある。一つの式の中に同順位の演算子が存在した場合、結合規則に基づいて優先評価される。たとえば、

```
a = 10;
b = 20;
a = b = 30;
```

では変数 a と b の値は共に 30 になる。これは演算子 ' = ' の結合規則が右結合的(右から左に結合する)となっているからである。このため式はまず右側の「 b = 30 」が実行され、そのあと「 a = b 」が実行される。

復習：算術演算子
－その２（優先順位と結合規則）

また

```
8 / 4 * 2
```

という式では `'/'` と `'*'` の優先順位は等しいけれど左結合であるため、

```
( 8 / 4 ) * 2
```

として働き、結果は `4` となる。

優先順位と結合規則

種類	演算子	結合法則
カッコ	( )	→
否定	!	←
乗除	* / %	→
加減	+ -	→
比較	< <= > >=	→
等値	== !=	→
論理積	&&	→
論理和	||	→
代入	= += -= *= /= %=	←
コンマ	,	→

for文による無限ループ　p.81

for文の特殊用法

p. 81

```
for(;;){                    /*  無限ループ  */
    ...
    if(条件) break;  /*  条件が真ならループを終了する  */
}
```

注意：　break文で脱出する.

p. 92

while文による繰り返し　p.82

while文は不定回数のループ処理を行う。

```
while(式){
    文
}
```

p. 82

- 式が真である間、文を繰り返し実行する
- **while**文は次のループに「入る条件」を判断する
- 式がはじめから偽の場合は一度も実行されない
- 文が単文の場合は{}を省略できる

while文の使用例

実行結果を予想せよ!

```
#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

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

while文による無限ループ　p.83

while文の特殊用法

```
while(1){                    /*  無限ループ  */
    ...
    if(条件) break;  /*  条件が真ならループを終了する  */
}
```

注意：　break文で脱出する.

p. 92

do-while文による繰り返し　p.84

```
do{
   文
}while(式);
```

p. 84

- 式が真なら処理を継続
- **do-while**では最低1回は文を実行する
- **do-while**文は現在のループを「継続する条件」を判断する
- 文が1つなら{}は省略できる
- 継続条件式の後ろに‘;’が必要

while	–前判定
do-while	–後判定（とりあえず1回は実行）

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

do-while文の使用例

配列ssのサイズは？
実行結果を予想せよ!

```
#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;
}
```

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
```

switch文による多方向分岐　p.87

p. 87

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

switch文による多方向分岐　p.87

- 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");
}
```

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;
}
```

実行結果を予想せよ!

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
```

switch文の定数式　p.88

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(P.30)，定数の式，記号定数(P.32)

正

```
case 5:   case 'A':  case 'B'+10:  case MAXVAL:
```

誤

```
case 5.6:  case a>10:  case "ABC":
```

switch文例2

```
#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;
}
```

実行結果を予想せよ!

switch文例2

```
#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
```

switch文例3

```
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;
}
```

実行結果を予想せよ!

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
```

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  */
```

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？

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*/
```

まとめ

- C言語の基礎
  - for文による無限ループ (p. 81)
  - while文 (p.82)
  - do-while文 (p. 84)
  - switch文 (p. 87)
  - キャスト演算子（p.62)