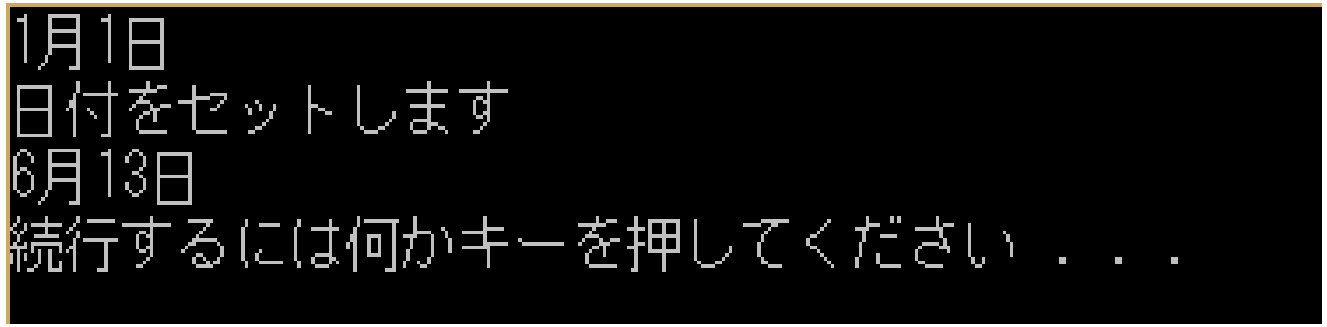
C言語講座　第6回

printf関数を用いて，日付を入力し，
その日付を自作関数set_dayを使って変えるプログラムを作成してください.



*ポインターを使うと……

関数で変更した値を，元の変数でも反映させられる。

```
//日付を変えるプログラム
#include <stdio.h>
void set_day(int m, int d);
int main(){
int month = 1;
int date = 1;
printf("%d月%d日¥n", month, date);
set_day(month, date);
printf("%d月%d日¥n", month, date);
return 0;
}
```

```
void set_day(int m, int d){
printf("日付をセットします¥n");
m = 6;
d = 13;
}
```

```
1月1日
日付をセットします
1月1日
続行するには何かキーを押してください . . .
```

１月１日を６月１３日に
変えたかったのに,
変わりません.

```
int month = 1;
int date = 1;
printf("%d月%d日¥n", month, date);
set_day(month, date);
printf("%d月%d日¥n", month, date);
return 0;
}

void set_day(int m, int d){
printf("日付をセットします¥n");
m = 6;
d = 13;
}
```

month,dateの値を
set_day関数に渡す

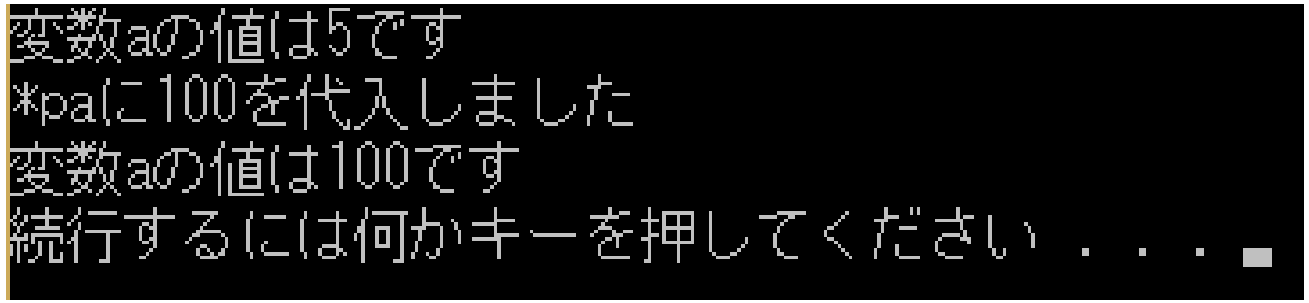
main関数からset_day関数が呼び出されたときに，　set_day関数にint型の変数m, d という変数が宣言され、それぞれ渡された値を代入します。

しかし、このm,dはmain関数のmonth,dateの値を代入しただけの別の変数なのです。そのため、このm,dを変えても、main関数の変数が入れ替わらないのです。（違う変数だからね。）

```
#include <stdio.h>

int main(){
int a;
int *pa;

 a = 5;
 pa = &a;

 printf("変数aの値は%dです¥n", a);

 *pa = 100;

 printf("*paに100を代入しました¥n");
 printf("変数aの値は%dです¥n", a);

 return 0;

}
```

ポインタpaを宣言

aのアドレスをpaに入れる



このことから,
*pa=a であり,
*pa=100; a=100;　といえます

さっきの日付のプログラムでは，同じものを
指していないため，日付を変えることが出来ませんでした.

でも，これを使い
monthと*m
dateと*d
で，同じにできそう！

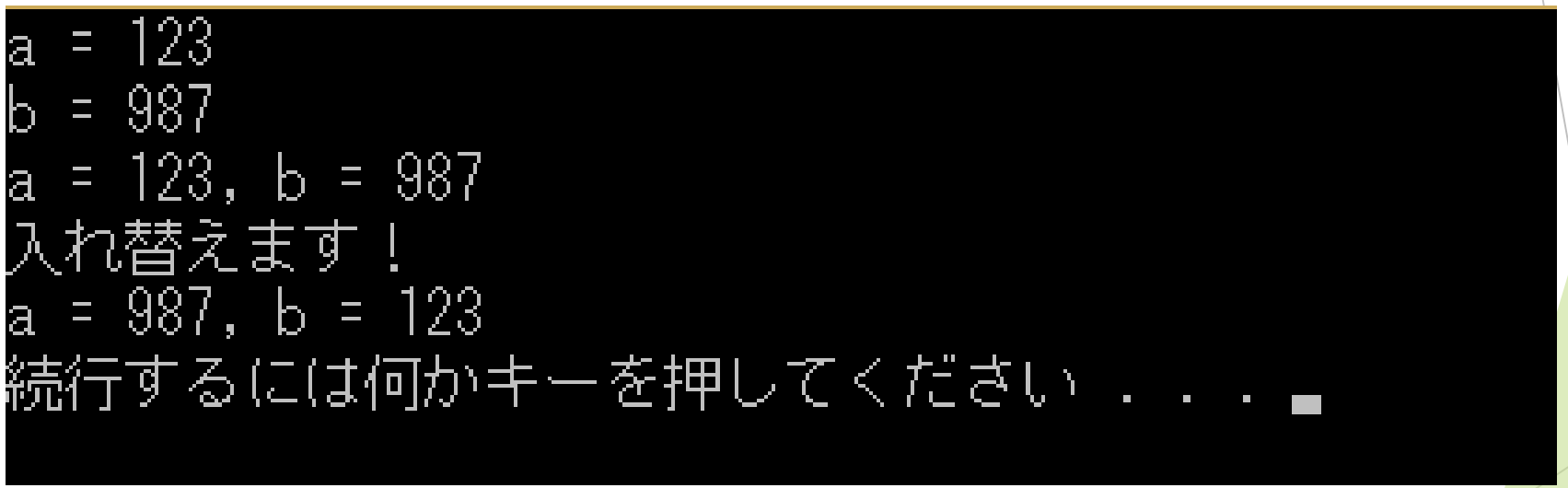
```
//日付を変えるプログラムその２
//赤字のところが変化した場所

#include <stdio.h>

void set_day(int *m, int *d);

int main(){
int month=1;
int date=1;

printf("%d月%d日¥n",month,date);
set_day(＆month, ＆date);

printf("%d月%d日¥n",month,date);

return 0;
}

void set_day(int *m, int *d){
printf("日付をセットします¥n");
*m=6;
*d=13;
}
```

できた！

void set_day(int* m, int* d)
↑関数の宣言部
引数がポインタ変数になっている
つまり、アドレスを受け取る
ポインタを使って，値を交換

set_day(&month, &date);
↑main文内の関数呼び出し文
&変数名 で変数のアドレスを表している
つまり、変数のアドレスを渡している

int型変数 a，bをscanf関数を用いて入力し、自作関数swap()を作成して値a,bを交換して表示してください。



```
#include<stdio.h>

void swap(int *x, int *y){
int tmp;
printf("入れ替えます！¥n");
tmp = *x;
*x = *y;
*y = tmp;
}
```

```
int main(void){
int a, b;

printf("a = ");
scanf_s("%d", &a);

printf("b = ");
scanf_s("%d", &b);

printf("a = %d, b = %d¥n", a, b);

swap(&a, &b);
printf("a = %d, b = %d¥n", a, b);
return 0;

}
```

5名のそれぞれの得点を20点満点とし、最高点と最低点の各1名を除いた、残り3名の得点の合計を求めてください。
ただし，
2つの数値のうち小さい方の値を戻り値とする関数 `lt(a,b)`
2つの数値のうち大きい方の値を戻り値とする関数 `gt(a,b)`
を作成し，使用してください.

```
得点を入力: 10
得点を入力: 11
得点を入力: 12
得点を入力: 13
得点を入力: 14
最高点: 14
最低点: 10
獲得点: 36
続行するには何かキーを押してください . . .
```

```
#include<stdio.h>

int input(void);
void output(int, int, int);
int gt(int, int);
int lt(int, int);

int main()
{
int a;
int max;
int min;
int sum;
int i;

max = 0;
 min = 20;
 sum = 0;
 i = 0;

 while (i < 5){
 a = input();
 max = gt(a, max);
 min = lt(a, min);
 sum += a;
 i++;

 }

  sum -= max + min;

 output(max, min, sum);

 return 0;

 }
```

```
/* キーボードから入力する関数 */
 int input(void)
 {
 int x;

 printf("得点を入力: ");
 scanf_s("%d", &x);

 return x;
  }

 /* 画面に出力する関数 */
 void output(int max, int min, int sum)
 {
  printf("最高点: %d¥n", max);
  printf("最低点: %d¥n", min);
  printf("獲得点: %d¥n", sum);

  }
```

```
/* 2つの値のうち大きい値を返す関数 */
int gt(int a, int b)
{
 int g;

 if (a > b){
 g = a;

}
else{
 g = b;

}

 return g;

 }
```

```
/* 2つの値のうち小さい値を返す関数 */
 int lt(int a, int b)
  {
int l;

if (a < b){
 l = a;

}
else{
 l = b;

}

 return l;
 }
```

アルファベットを何文字かずらした暗号をシーザー暗号といいます。
例： HAL → IBM **（すべてを1文字後ろにずらした場合）**

文字列をN文字後ろに置換したシーザー暗号に変換する関数 zurasu**と**
元の文字列に戻す関数 modosu**を作成してください。**

ずらすのはアルファベットのみとし、大文字は大文字、小文字は小文字、Z の次は A （z の次は a ）とします.

```
文字列を入力してください:HAL
暗号文=> IBM
平文=> HAL
続行するには何かキーを押してください . . .
```

ヒント

```
#include <stdio.h>

 void zurasu(char *, int);
 void modosu(char *, int);

int main(void)
{
char str[1024];

printf("文字列を入力してください:");
Scanf_s("%s", str);

 zurasu(str, 1);
 printf("暗号文=> %s¥n", str);

 modosu(str, 1);
 printf("平文=> %s ¥n", str);

return 0;
 }
```

```
void zurasu(char *p, int n){

ここに，zurasuの処理を記述する

}

void modosu(char *p, int n){

ここに，modosuの処理を記述する

}
```

```
void zurasu(char *p, int n)
{
while (*p != '¥0') {
  if (*p >= 'a' && *p <= 'z') {
      *p += n % 26;
  if (*p>'z') {
      *p = *p - 26;

}

}
else if (*p >= 'A' && *p <= 'Z') {
      *p += n % 26;
if (*p>'Z') {
      *p = *p - 26;

   }

}
 ++p;

}
}
```

```
void modosu(char *p, int n)
{
while (*p != '¥0') {
    if (*p >= 'a' && *p <= 'z') {
        *p -= n % 26;
    if (*p<'a') {
        *p = *p + 26;

}

}
      else if (*p >= 'A' && *p <= 'Z') {
           *p -= n % 26;
       if (*p<'A') {
            *p = *p + 26;

}

}
++p;

}
}
```