Training 10
プリプロセッサ

株式会社イーシーエス　出版事業推進委員会

Lesson1　マクロ置換

Point◆◇マクロ置換を理解しよう!!

マクロ置換の機能により、文字列の置き換えをすることが出来ます。
プログラムの可読性と保守性（メンテナンス性）を高めることができるため、よく用いられます。
マクロ置換で値を定義しておけば、マクロの値を変更するだけで、同じマクロを使用したすべての箇所が変更ができるので便利です。

【問題1】　次のプログラムが実行されたとき、x に以下の値が入力された場合の、最終的な y の値を答えなさい。

```
#include <stdio.h>
#define  MAX  20
#define  MIN  10

void main(void)
{
     int x,y;
     scanf("%d",&x);

     if( x > MAX)
     {
          y = MAX;
     }
     else if( x < MIN )
     {
          y = MIN;
     }
     else
     {
          y = x;
     }
}
```

①　x = 9
②　x = 10
③　x = 11
④　x = 19
⑤　x = 20
⑥　x = 21

【問題2】　次の配列 str の値が、文字列「"abcdef"」となるように□部分の空欄を埋めなさい。

```
#define  ①    ②

const char str[] = "abc" MYSTR;
```

【問題3】　次の□部分を埋め、プログラムを完成させなさい。

【処理内容】
事前定義マクロを使い、コメントのとおりに動作する。

```
#include <stdio.h>

void main(void)
{
     /* ソースファイルのパス名を出力  */
     printf("  ①   = %s¥n",   ①   );

     /* 現在のソースファイルの行番号を出力  */
     printf("  ②   = %d¥n",   ②   );

     /* ソースファイルをコンパイルした日付を出力  */
     printf("  ③   = %s¥n",   ③   );

     /* ソースファイルをコンパイルした時刻を出力  */
     printf("  ④   = %s¥n",   ④   );
}
```

Lesson2　前処理制御

Point◆◇前処理制御を理解しよう!!

別ファイルの内容の読み込むインクルードや、不要なソースコードを読み飛ばすことができる制御コマンドが前処理制御です。
これらは便利なツールですので、しっかりと理解しておきましょう。

【問題1】　次の□部分を埋め、プログラムを完成させなさい。

【処理内容】
x の値を求める。(3 つのファイルは同じ階層のフォルダに存在するものとする。)

main.c

```
#include   ①

void main(void)
{
    int x;
    x = 10;

    x = keisan(x);
}
```

keisan.h

```
int keisan(int);
```

keisan.c

```
int keisan(int z)
{
    z = ( z + 20 ) * 100;
    return z;
}
```

【問題2】　次の□部分を埋め、プログラムを完成させなさい。

【処理内容】
標準ライブラリの関数 sqrt を使用し、x の値を求める。(必要な標準ヘッダファイルは「math.h」とする)

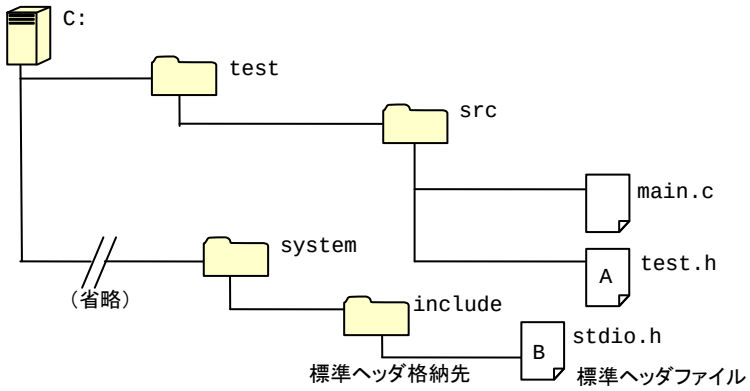
```
#include   ①

void main(void)
{
    double   x;

    x = sqrt(10.5);
}
```

【問題3】 下図のような階層でファイルが存在し、**main.c** に作成した **test.h** と、標準ヘッダファイルの **stdio.h** を読み込みたい場合はどのように記述するか。次の□部分を埋めて答えなさい。

test.h はカレントディレクトリを標準の場所として探す記述をし、stdio.h は直接、標準の場所(標準ヘッダ格納先)のみを探す記述とする



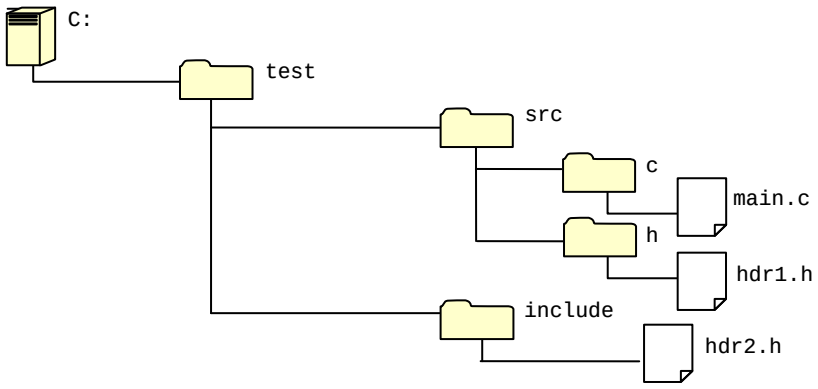
main.c

```
/* test.h を読み込む */
#include  ①

/*stdio.h を読み込む */
#include  ②

void main(void)
{
    (略)
}
```

【問題4】 下図のような階層でファイルが存在し、**main.c** に **hdr1.h** と **hdr2.h** を読み込みたい場合はどのように記述するか。次の□部分を相対パス指定で埋めて答えなさい。



main.c

```
/* hdr1.h を読み込む */
#include  ①

/* hdr2.h を読み込む */
#include  ②

void main(void)
{
    (略)
}
```

【問題5】 次のプログラムを実行した時の、最終的な y の値を答えなさい。

```
void main(void)
{
    int x,y;
    x = 0;
    y = 1;
#if 1
    x = 2;
    y = x + 2;
#else
    y = x + 3;
#endif
    y = y * 2;
}
```

【問題6】 次のプログラムを実行した時の、最終的な y の値を答えなさい。

```
#define COND  0
void main(void)
{
    int x,y;
    x = 1;
    y = 1;
#if COND > 0
    x = x + 2;
#elif COND == 0
    x = x + 3;
#else
    x = x + 5;
#endif
    y = x * 2;
}
```

【問題7】 次のプログラムを実行した時の、最終的な y の値を答えなさい。

```
#define BBB
void main(void)
{
    int x,y;
    x = 1;
    y = 1;

#ifdef AAA
#ifndef BBB
    x = x + 1;
#else
    x = x + 2;
#endif
#else
#ifndef BBB
    x = x + 10;
#else
    x = x + 15;
#endif
#endif
    y = x * 2;
}
```

Lesson3　関数マクロ

Point◆◇関数マクロを理解しよう!!

マクロに関数を定義することもできます。この関数マクロは引数をつけて定義することもできます。
関数マクロを定義するときには、結果が意図しない値にならないように「( )」を適切につけましょう。

【問題1】　次のプログラムが実行されたときの、最終的な w、x、y、z の値を答えなさい。

```
#include <stdio.h>

#define  KAKE_A(a) (a * 10)
#define  KAKE_B(b) ((b) * 10)

void main(void)
{
    int w,x,y,z;

    w = KAKE_A(6);
    x = KAKE_B(6);
    y = KAKE_A(6 + 2);
    z = KAKE_B(6 + 2);
}
```

① w
② x
③ y
④ z

【問題2】　次のプログラムを読み、以下の問に答えなさい。(各ファイルは同じ階層にある)

type.h

```
#ifndef __TYPE_H__
#define __TYPE_H__

/* 基本型定義 */
typedef signed char       S1;
typedef unsigned char     U1;
typedef signed short      S2;
typedef unsigned short    U2;
typedef signed long       S4;
typedef unsigned long     U4;
typedef void              VD;

#endif
```

main.h

```
#ifndef __MAIN_H__
#define __MAIN_H__

#define NUM  7

const S1 MAX = round( 2.3 );
const S1 MIN = round( -2.6 );

const S1 num_table[NUM] = {
CAL_A( -6.0 ) , CAL_A( -1.4 ) , CAL_A( -2.1 ) ,
CAL_B( 26.4 ) , CAL_B( 15.2 ) ,
CAL_C(-1.1) , CAL_C( 0.9 )
};

#undef CAL_A
#undef CAL_B
#undef CAL_C

#endif
```

routine.h

```
#ifndef __ROUTINE_H__
#define __ROUTINE_H__

#define round(i) ( ( S1 )( ( ( i ) >= 0.0 ) ? ( ( i ) + 0.5 ) : ( ( i ) - 0.5 ) ) )

#define CAL_A(a) ( ( S1 )( ( a ) * 2.0 ) )
#define CAL_B(b) ( ( S1 )( ( ( b ) / 2.0 ) - 5.4 ) )
#define CAL_C(c) ( ( S1 )( ( c ) + 1.1 ) )

#endif
```

main.c

```
#include <stdio.h>
#include "type.h"
#include "routine.h"
#include "main.h"

VD main(VD)
{
    S1 i;
    S1 max_over;
    S1 min_under;

    max_over = 0;
    min_under = 0;

    for( i = 0 ; i < NUM ; i++)
    {
        if( num_table[i] >= MAX )
        {
            max_over++;
        }
        else if( num_table[i] <= MIN )
        {
            min_under++;
        }
    }
    printf("max_over の値は%d になります。¥n", max_over);
    printf("min_under の値は%d になります。¥n", min_under);
}
```

① MAX、MIN の値を答えなさい。

② num_table の値を全て答えなさい。

③ printf で出力される max_over、min_under の値を答えなさい。

Lesson4　マクロの応用

Point◆◇マクロを使ったソースを読めるようにしよう!!

マクロを使いスイッチを作ることにより、スイッチの切り替えで違うプログラムを実行することができます。
移植性を持たせたいときなどに効果を発揮します。

【問題1】 以下のプログラムを読み問いに答えなさい。(各ファイルは同じ階層にある)

main.h

```
/* 基本型定義 */
typedef signed char        S1;
typedef unsigned char      U1;
typedef signed short       S2;
typedef unsigned short     U2;
typedef signed long        S4;
typedef unsigned long      U4;
typedef void               VD;

/* マクロ定義 */
#define SWITCH1 1
#define SWITCH2 2
#define SWITCH3 3
#define SWITCH4 4

#define SIMUKE  A
```

main.c

```
#include <stdio.h>
#include "main.h"

VD main(VD)
{
    U2 in_a;
    U2 in_b;
    U2 ans;
    U1 str[5] = { 7,3,4,2,0 };
    U1 i;
    U1 j;

    /* 初期化 */
    in_a = 2;
    in_b = 4;
    ans  = 0;

#if SIMUKE == SWITCH2
    for( i = 0; i < 4; i++ )
    {
        ans += in_a;
    }
#endif
#if SIMUKE == SWITCH1
    ans = in_a + in_b;
#endif
#if SIMUKE == SWITCH2
    for( i = 0; i < 4; i++ )
    {
        ans -= in_a;
    }
#endif
#if SIMUKE == SWITCH1
    for( i = 0; i < 2; i++ )
    {
        ans += in_b;
        for( j = 0; j < 3; j++ )
        {
            ans += in_a;
```

```
        }
    }
#else
    for( i = 0; i < 5; i++ )
    {
        str[i] = ( ans + in_a );
    }
#endif

#if SIMUKE == SWITCH3
    ans += in_b;
    ans *= in_a;
#elif SIMUKE == SWITCH4
    ans %= 2;
    for( i = 0; i < 3; i++ )
    {
       ans += ( in_b + in_a );
    }
#endif
    if( ( ans >= 100 ) || ( str[3] >= 5 ) )
    {
      printf(" ansの値は%dです。", ( ans + str[i] ) );
    }
    else
    {
        printf(" ansの値は%dです。",  ans );
    }
}
```

① [A] をSWITCH1にしたときの printf の出力結果を答えなさい。

② [A] をSWITCH2にしたときの printf の出力結果を答えなさい。

③ [A] をSWITCH3にしたときの printf の出力結果を答えなさい。

④ [A] をSWITCH4にしたときの printf の出力結果を答えなさい。

解答

Training10 プリプロセッサ

Lesson1 マクロ置換

問題 1	①10　②10　③11　④19 ⑤20　⑥20
問題 2	①MYSTR　②"def"
問題 3	①__FILE__　②__LINE__ ③__DATE__　④__TIME__

Lesson2 前処理制御

問題 1	①"keisan.h"
問題 2	①<math.h>
問題 3	①"test.h"　②<stdio.h>
問題 4	①"../h/hdr1.h"　※(Windows 上の多くのコンパイラでは "..¥h¥hdr1.h" でも可) ②"../../include/hdr2.h"　※(Windows 上の多くのコンパイラでは "..¥..¥include¥hdr2.h" でも可)
問題 5	8
問題 6	8
問題 7	32

【解説】問題 7
このプログラムでは、#define で定義されているのは BBB だけです。
つまり AAA は定義されてなく、BBB は定義されている状態です。
そのため、18 行目で x の値が 16 となり、21 行目で y は
16×2=32 となっている。

Lesson3 関数マクロ

問題 1	①60　②60　③26　④80
問題 2	①MAX : 2 MIN : -3 ②num_table[] = { -12, -2, -4, 7, 2, 0, 2}; ③max_over: 3 min_under : 2

【解説】問題 2
整数の定義について
プログラムでは整数の定義を0.0、-6.0などのようにあえて小数表記をしています。これにより、数値を double 型として定義できます。

関数マクロ round(i)について
((i) >= 0.0) ? ((i) + 0.5) : ((i) - 0.5))
は条件と、その条件が真の場合、偽の場合の処理を 1 行で表したものです。
条件は '? 'の前の(i) >= 0.0 です。
真の場合の処理は'? 'の後の(i) + 0.5 です。
偽の場合の処理は': 'の後の(i) - 0.5 です。
そしてその結果を S1(signed char)でキャストをしています。
この関数マクロでは引数が正の数なら0.5を足して、負の数なら0.5を引いて小数点以下を切り捨て、整数にしています。(四捨五入の処理)

①MAX、MIN の初期化方法
main.h 内で MAX、MIN は変数宣言で関数マクロを呼んで初期化を行っています。

②num_table の設定
関数マクロ CAL_A(a)、CAL_B(b)、CAL_C(c)を用いて num_table を作成しています。引数として受け取った値をそれぞれ計算して、その値を S1 でキャストをしています。

③max_over、min_under の算出
main 関数で num_table の値の中で、MAX の値以上になる数、MIN の値以下になる数を数えています。

Lesson4 マクロの応用

問題 1	ans の値は 26 になります。
問題 2	ans の値は 0 になります。
問題 3	ans の値は 8 になります。
問題 4	ans の値は 18 になります。