関数の例

- (1) 関数名が hoge である関数を作れ。戻り値、引数はなし(void)でよい。関数の中身は省略で良い。

```
void hoge(){  
    (省略)  
}
```

- (2) 関数名が hoge であり、関数の中身が printf("Hello\n"); である関数を作成せよ。戻り値、引数はなし(void)でよい。

```
void hoge(){  
    printf("Hello");  
}
```

- (3) 関数名が hoge で、引数が int 型 1 個である関数を作成せよ。戻り値はなし(void)でよい。関数の中身は省略で良い。引数の変数名は自由に決めて良い。

```
void hoge(int a){  
    (省略)  
}
```

- (4) 関数名が hoge で、引数が int 型の a と int 型の b の 2 個である関数を作成せよ。戻り値はなし(void)でよい。関数の中身は省略で良い。

```
void hoge(int a, int b){  
    (省略)  
}
```

- (5) 関数名が hoge で、引数が int 型の a と int 型の b の 2 個で、戻り値が int 型で 3 である関数を作成せよ。

```
int hoge(int a, int b){  
    return 3;  
}
```

- (6) 関数名が abc で、引数が int a と int b の 2 個で、戻り値がない(void である)関数を作成せよ。関数の中身は a と b の値を printf で表示するものとせよ。

```
void abc(int a, int b){  
    printf("%d %d\n", a, b);  
}
```

- (7) 関数名が hoge で、引数が int 型 1 個で、戻り値として引数で受けとった数の 2 倍の値を返す関数を作成せよ。例えば、引数として 3 を与えて関数を呼び出した場合、戻り値として 6 が返される。戻り値の型は int 型とせよ。

```
int hoge(int x){  
    return x*2;  
}
```

- (8) 関数名が bar で、引数が int 型 a と int 型 b の 2 個で、戻り値が int 型で a と b の和である関数を作成せよ。

```
int bar(int a, int b){  
    return a+b;  
}
```

- (9) 関数名が foo で、引数が int x と int y の 2 個で、戻り値が int 型である関数を作成せよ。関数の中身は、x と y を printf により表示して、x と y の積を戻り値として返すものとせよ。

```
int foo(int x, int y){  
    printf("%d %d\n", x, y);  
    return x*y;  
}
```

- (10) 関数名が abs で、引数が double 型 1 個で、戻り値として引数の絶対値を返す関数を作成せよ。戻り値の型は double 型とせよ。

```
double abs(double x){  
    if( 0<=x ){  
        return x;  
    } else {  
        return -x;  
    }  
}
```

(1) 以下はどんな関数？

```
void hoge(){
    (省略)
}
```

戻り値が void 型(なし)で, 関数名が hoge で, 引数がない関数. 関数の中身は省略されているので不明.

(2) 以下はどんな関数？

```
int piyo(){
    (省略)
}
```

戻り値の型が int 型で, 関数名が piyo で, 引数がない関数. 関数の中身は省略されているので不明. 戻り値の型は int 型と分かるが, (関数の中身が省略されているので)戻り値の値は不明.

(3) 以下はどんな関数？

```
void fuga(int x){
    (省略)
}
```

戻り値がなく(void型)で, 関数名が fuga で, 引数が int 型の x の 1 個である関数. 関数の中身は省略されているので不明.

(4) 以下はどんな関数？

```
void bar(int a, int b){
    (省略)
}
```

戻り値がなく(void型)で, 関数名が bar で, 引数が int 型の a と int 型の b の 2 個である関数. 関数の中身は省略されているので不明.

(5) 以下はどんな関数？

```
void foo(int x){
    printf("%d\n", x);
}
```

戻り値がなく(void型)で, 関数名が foo で, 引数が int 型の x の 1 個である関数. 関数の中身は, 引数で受けとった int x の値を printf() により表示している.

(6) 以下はどんな関数？

```
int hoge(double d,int i ){
    (省略)
}
```

戻り値の型は int 型. 関数の中身が省略されているので戻り値の値は不明. 関数名が hoge. 引数は double 型の d と, int 型の i の 2 個.

(7) 以下はどんな関数？

```
int piyo(){
    return 3;
}
```

戻り値は, 型が int 型で, 値が 3. 関数名は piyo. 引数はない. 関数の中身は"3 を return する"のみである.

(8) 以下はどんな関数？

```
double piyo(){
    return 3.3;
}
```

戻り値は, 型が double 型で, 値が 3.3 である. 関数名は piyo. 引数はない. 関数の中身は"3.3 を return する"のみである.

(9) 以下はどんな関数？

```
int inc(int x){
    return x+1;
}
```

関数名 inc. 引数は int 型の x の 1 個. 戻り値は, 型が int 型で, 値が引数で受けとった数+1.

(10) 以下はどんな関数？

```
int abs(int x){
    if(x<0){
        return -x;
    } else {
        return x;
    }
}
```

関数名 abs, 引数 int 型. 戻り値は int 型で, 引数が負の場合は -x, 0 か正の場合 x.

(1) 関数 `int inc(int x)` の呼び出し

```
int inc(int x){
    return x+1;
}
void main(){
    int i;
    i = inc(3);
    printf("i=%d\n", i);
}
```

実行結果 `i=4`

関数 `inc` は、引数が `int` 型 1 個なので、それに併せて呼び出す必要がある。この例では、`int` 型である 3 を引数に指定し呼び出している。例えば、`inc()` といった引数なしの呼び出しや、`inc(3,4)` といった引数 2 個の呼び出しは NG である。戻り値として `int` 型の値が返されるので、それを `int` 型変数 `i` に代入している。

(2) 関数 `double add(double x, double y)` の呼び出し

```
double add(double x, double y){
    return x+y;
}
void main(){
    double d;
    d = add( 1.2 , 3.4 );
    printf("d=%lf\n", d);
}
```

実行結果 `d=4.600000`

関数 `add` は、引数が `double x` と `double y` の 2 個なので、小数の 1.2 と小数の 3.4 を引数に指定し呼び出している。`add()` といった引数なしの呼び出しや、`inc(1.2)` といった引数 1 個の呼び出しは NG である。戻り値として `double` 型の値が返されるので、それを `double` 型変数 `d` に代入している。

関数の配置場所

```
void hoge(int x){
    printf("%d\n", x);
}
void piyo(int n){
    hoge(n*10);
}
void main(){
    hoge(3);
    piyo(2);
}
```

`void main()` も、`void hoge(int x)` も、`int piyo(int n)` も、全て関数である。よって、関係は対等である。`main()` の中に `hoge()` があるのは NG であり、`hoge()` の中に `main()` があるのは NG である。関数 `piyo()` の中で関数 `hoge()` を呼び出しているので、`piyo()` よりも `hoge()` を先に定義する。

正しくない例(A)

```
void main(){
    void hoge(int x){
        printf("%d\n", x);
    }
    hoge(3);
}
```

`main()` の中に、`hoge()` があるので NG.

正しくない例(B)

```
void hoge(int x){
    printf("%d\n", x);
}
void main(){
    hoge(3);
}
```

`hoge()` の中に、`main()` があるので NG.

ヒント

A) "戻り値なし"と"戻り値が `void`"は同意.