関数, subprogram

きまった仕事 (ルーチン作業) を請け負ってくれる関数についてです．これで，プログラミング言語の文法について一通り終わりです．後は，自分の好きなように思考 (試行) してください．

基本形

先ず，基本的な関数呼び出しを示します．

```
#include <stdio.h>
int func(int ii);      //(0)

int main(void){
  int i=4,j;
  j=func(i);        //(1) (5)
  printf("%d\n",j);
  return 0;
}

int func(int ii){     //(2)
  int n;
  n = ii*ii;        //(3)
  return n;         //(4)
}
```

ここで，(0) にあるように，main の前であらかじめ「これこれの」関数が使われますよという予告をしておきます．これをプロトタイプ宣言と言います.「これこれ」の中身は，戻り値が int 型で，名前が func で，引数が int 型の関数です.

順番に何が起こったかを記すと

1. 関数の呼び出し
2. 実引数 (i) の値を，仮引数 (ii) にコピー
3. 関数の中身を実行
4. 結果を return で返す.
5. j に代入される.

main, void

main も関数の一種である事がわかりますね．main の引数 void は何も型および値がないということを明示的に書くために使います．戻り値のない void 型関数というのもあります.

配列の受け渡し

配列を受け渡す場合は，以下の通り.

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

```
void print_array(int aa[], int n);

int main(void){
  int a[4]={1,2,3,4},n=4;
  print_array(a,n);
  return 0;
}

void print_array(int aa[], int n){
  int i;
  for (i=0;i<n;i++){
    printf("%3d",aa[i]);
  }
  printf("\n");
}
```

練習問題

1. Newton-Raphson 法をもちいて，$x - \cos x = 0$ の解を求めよ．ただし，$f'(x) = 1 + \sin x$.

   Newton-Raphson 法は適当な値 x_0 から出発して，

   $$x1 = x0 - \frac{f(x0)}{f'(x0)} \tag{1}$$

   という操作を順次繰り返し，解に近づけていく．これは $y = f(x)$ のグラフに接線をひいて x 軸との交点を次の近似値とする事に相当する．

   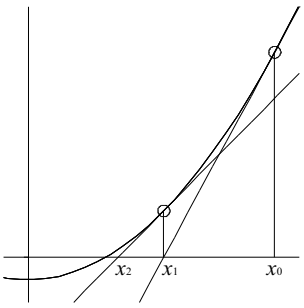

   図 1: Newton-Raphson 法の模式図.

   for-loop で 3 回ほど回して $x0, x1, f(x)$ を出力してみよ．最後は「ある小さな値 (eps=1.0e-10 など) を仮定して，前回の近似値との差がこの値以下になれば終了する」というのが常とう手段 (収束判定条件).

2. 二つの位置座標

   ```
   0.0 0.0
   1.0 1.0
   ```

   から距離を求める関数を作れ.

   次に，4 つの位置座標

```
0.0 0.0
1.0 1.0
1.0 0.0
0.0 1.0
```

を p[4][2] に読み込んで，{0,1,2,3,0} と巡る距離を求めよ．引数の次元によっては

```
error: invalid use of array with unspecified bounds
```

などの error がでる．以下の「多次元配列の受け渡し」を参照せよ．

おまけ

多次元配列の受け渡し

関数へ多次元配列を受け渡すには，以下のように配列の後ろ側の要素数がいくつあるか (M=2) を明示しておく必要がある．

```
#include <stdio.h>
enum {M=2};

int tmp(int d[][M]);

int main(void){
  int d[3][M]={1,2,3,4,5,6};
  tmp(d);
  return 0;
}

int tmp(int d[][M]){
  int i,j;
  for (i=0;i<3;i++){
    for (j=0;j<M;j++){
      printf("%3d",d[i][j]);
    }
    printf("\n");
  }
  return 0;
}
```

関数の引数の受け渡し

関数の引数の受け渡しはポインターを理解する必要がある．初級プログラムでは意識しなくていいのでパス．以下は必要になったときに読んでください．

関数に変数を渡すときのデフォルト仕様が単なる変数と配列で異なる．変数では上述の (2) で記した通り，実引数の値を仮引数にコピーする**値渡し**が使われる．これに対して，配列では配列のアドレスを渡す**参照渡し**が使われる．

この結果，関数の中で単なる変数の値を変更しても実引数の値は変わらないが，配列の値を変更すると実引数の値も変わる．

参照渡しでのプログラムを示しておく．i についている*や&を取ってみて出力結果がどのように違うかを確かめてみよ．

```
#include <stdio.h>

int func(int *i);

int main(void){
  int i=4,j;
  j=func(&i);
  printf("%d %d\n",i,j);
  return 0;
}

int func(int *i){
  int n;
  *i = *i +1;
  n = *i * *i;
  return n;
}
```