2015/06/03

第５回 Ｃ言語勉強会

関数

加納 徹

for 文の復習

```
for (初期化; 条件式; 次の一歩) {
    繰り返す処理内容
    （計算や文字の表示等）
}
```

プログラムの流れ

①初期化 ⇨ ② 条件式 ⇨ ③ 処理 ⇨ ④ 次の一歩

条件式を**満たす間**、処理を繰り返す命令

for 文の復習

10 回文章を表示するプログラム

```c
#include <stdio.h>

int main(void) {
	int i;
	for (i = 1; i <= 10; i++) {
		printf("%d 回目の Hello, World!¥n", i);
	}
	return 0;
}
```

10 までの和を計算するプログラム

```c
#include <stdio.h>

int main(void) {
	int i, sum = 0;
	for (i = 1; i <= 10; i++)
		sum += i;
	printf("1 から 10 までの和は %d¥n", sum);
	return 0;
}
```

if文やfor文は、処理が一行なら中括弧を省略できる

本日の流れ

1. 関数って何？
2. 関数の作り方
3. 練習問題
4. まとめ
5. おまけ問題

１．関数って何？

関数って何？

引数
（入力）

引数

戻り値
（出力）

戻り値

関数
（計算や処理）

数学の関数は、決められた **計算** をするもの
プログラミングの関数は、決められた **仕事** をするもの

関数って何？

引数
（入力）

$x = 2$

関数

$f(x) = x^2 + 3x$

戻り値
（出力）

10

一度関数を作れば何度でも再利用することができる。
長いコードを書く場合には必要不可欠。

２．関数の作り方

$f(x) = x^2 + 3x$ の計算

```c
#include <stdio.h>

int main(void){
    int ans;

    // f(2) の計算
    ans = 2 * 2 + 3 * 2;
    printf("f(2) = %d¥n", ans);

    // f(-1) の計算
    ans = (-1) * (-1) + 3 * (-1);
    printf("f(-1) = %d¥n", ans);

    return 0;
}
```

ここを関数にしたい

関数の作り方①

戻り値の型　関数の名前　引数

```
int calc(int x) {
    関数の処理内容
    （計算や文字の表示等）
}
```

戻り値や引数が無い場合は、void（空っぽの意味）と記入

関数の作り方②

計算結果用の変数

```
int calc(int x) {
    int res;
    res = x * x + 3 * x;
    return res;
}
```

計算内容の記述

return で呼び出し元に結果を返す

引数として受け取った値は x に格納され、関数内で自由に使える

関数の作り方③

```
#include <stdio.h>

int calc(int x);

int main(void) {
    int ans;
    ans = calc(2);
    printf("f(2) = %d¥n", ans);
    return 0;
}

int calc(int x) {
    int res;
    res = x * x + 3 * x;
    return res;
}
```

プロトタイプ宣言
この関数を使うということを
事前にプログラムに教える

関数の呼び出し

関数の作り方④（まとめ）

```c
#include <stdio.h>

int calc(int x);

int main(void){
    int ans;

    // f(2) の計算
    ans = calc(2);
    printf("f(2) = %d¥n", ans);

    // f(-1) の計算
    ans = calc(-1);
    printf("f(-1) = %d¥n", ans);

    return 0;
}

int calc(int x) {
    int res;
    res = x * x + 3 * x;
    return res;
}
```

関数のプロトタイプ宣言

関数の呼び出し

関数の内容（$x^2 + 3x$ を返す関数）

計算結果を返す

コードを短く美しく

```c
#include <stdio.h>

int calc(int x);

int main(void){
    // f(2) の計算
    printf("f(2) = %d¥n", func(2));

    // f(-1) の計算
    printf("f(-1) = %d¥n", func(-1));

    return 0;
}

int calc(int x) {
    return x * x + 3 * x;
}
```

関数の呼び出し

よりシンプルな記述

3. 練習問題

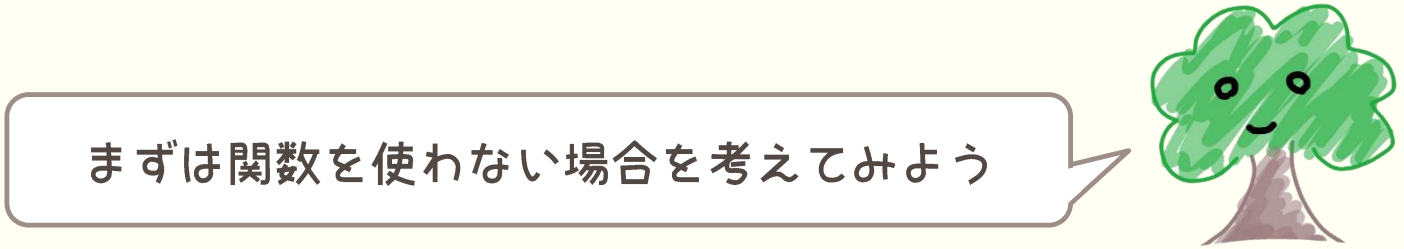
練習問題

問題１

int 型の引数 n を受け取って、１ ～ n までの整数の和を戻り値として返す関数 calc_sum1 を作りなさい。
（次のページのコードを使っても良い）

問題２

int 型の引数 m と n を受け取って、m ～ n までの整数の和を戻り値として返す関数 calc_sum2 を作りなさい。



関数を使わない場合

```
#include <stdio.h>

int main(void){
	int i, sum;
	// 1 から 10 までの和
	sum = 0;
	for (i = 1; i <= 10; i++) {
		sum += i;
	}
	printf("1 から 10 までの和は、%d¥n", sum);
	// 1 から 100 までの和
	sum = 0;
	for (i = 1; i <= 100; i++) {
		sum += i;
	}
	printf("1 から 100 までの和は、%d¥n", sum);

	return 0;
}
```

ここを関数にしたい

問題１　解答例

```c
#include <stdio.h>

int calc_sum1(int n);

int main(void){
	// 1 から 10 までの和
	printf("1 から 10 までの和は、%d¥n", calc_sum1(10));
	// 1 から 100 までの和
	printf("1 から 100 までの和は、%d¥n", calc_sum1(100));
	return 0;
}

int calc_sum1(int n) {
	int i, sum = 0;
	for (i = 1; i <= n; i++) {
		sum += i;
	}
	return sum;
}
```

プロトタイプ宣言

関数の呼び出し

1 から n までの和を計算して結果を返す関数

問題２ 解答例

```c
#include <stdio.h>

int calc_sum2(int m, int n);

int main(void){
	// 5 から 10 までの和
	printf("5 から 10 までの和は、%d¥n", calc_sum2(5, 10));
	// 10 から 100 までの和
	printf("10 から 100 までの和は、%d¥n", calc_sum2(10, 100));
	return 0;
}

int calc_sum2(int m, int n) {
	int i, sum = 0;
	for (i = m; i <= n; i++) {
		sum += i;
	}
	return sum;
}
```

プロトタイプ宣言

引数を2つ与えて呼び出し

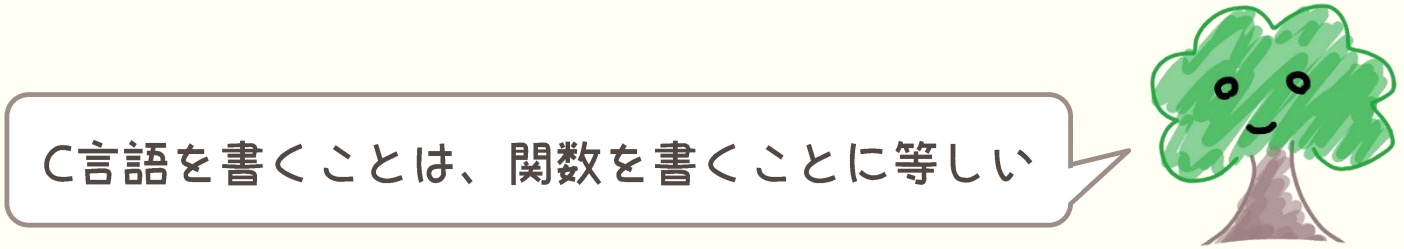
引数として m と n の2つを受け取る

m から n までの和を計算して結果を返す関数

４．まとめ

関数のまとめ

- 関数とは、決められた **仕事** をするもの。
- 関数は、**[戻り値の型] [関数名] ([引数])** の形で定義される。
- 関数を使う際は、必ず **プロトタイプ宣言** をする。
- 戻り値がある場合は、**return 文** で記述する。
- 戻り値や引数が無い場合は、定義の際に **void** を記述する。



５．おまけ問題（再帰関数）

おまけ問題

問題１

int 型の引数 n を受け取って、その絶対値を返す関数 calc_abs を作りなさい。

問題２-A

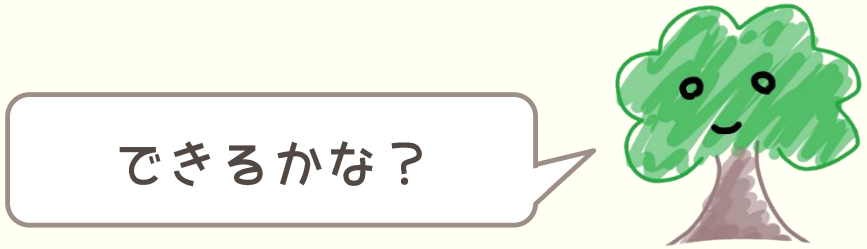
int 型の引数 n を受け取って、n! を返す関数 calc_factを作りなさい。
また、n に様々な値を入れて結果を検証しなさい。

問題２-B

int 型の引数 n と k を受け取って、${}_nC_k \left(= \frac{n!}{k!(n-k)!}\right)$ を返す関数 calc_comb を作りなさい。ただし、関数 calc_fact を利用すること。

問題２-C

関数 calc_fact を、 for ループを使わずに実装しなさい。



おまけ問題１　解答例

```c
#include <stdio.h>

int calc_abs(int n);

int main(void) {
        int n;
        printf("n = ");
        scanf("%d", &n);
        printf("| n | = %d¥n", calc_abs(n));
        return 0;
}

int calc_abs(int n) {
        if (n < 0)
                return -n;
        else
                return n;
}
```

プロトタイプ宣言

関数の呼び出し

n が負の数だった場合
マイナスをつけて返す

おまけ問題２-Ａ 解答例

```c
#include <stdio.h>

int calc_fact(int n);

int main(void) {
	int n;
	printf("n = ");
	scanf("%d", &n);
	printf("n! = %d¥n", calc_fact(n));
	return 0;
}

int calc_fact(int n) {
	int i, res = 1;
	for (i = 1; i <= n; i++) {
		res *= i;
	}
	return res;
}
```

プロトタイプ宣言

関数の呼び出し

1 から n までの積を
計算して返す関数

おまけ問題２-Ｂ 解答例

```c
#include <stdio.h>

int calc_fact(int n);
int calc_comb(int n, int k);

int main(void) {
        int n, k;
        printf("n = ");
        scanf("%d", &n);
        printf("k = ");
        scanf("%d", &k);
        printf("nCk = %d¥n", calc_comb(n, k));
        return 0;
}

int calc_fact(int n) {
        int i, res = 1;
        for (i = 1; i <= n; i++) res *= i;
        return res;
}

int calc_comb(int n, int k) {
        return calc_fact(n) / (calc_fact(k) * calc_fact(n - k));
}
```

プロトタイプ宣言

関数の呼び出し

関数の中から calc_fact を
呼び出し、nCkを計算する関数

おまけ問題２-Ｃ 解答例

```c
#include <stdio.h>

int calc_fact(int n);

int main(void) {
        int n, k;
        printf("n = ");
        scanf("%d", &n);
        printf("n! = %d¥n", calc_fact(n));
        return 0;
}

int calc_fact(int n) {
        if (n == 0)
                return 1;
        else
                return n * calc_fact(n - 1);
}
```

プロトタイプ宣言

関数の呼び出し

自分自身を繰り返し呼び出すことで、for 文を使わずに階乗を計算する関数（再帰関数）

おわり