初心者のための C言語講座

#2: printf(), scanf()

#2-1）printf()

さて、今回から段々プログラミングっぽくなっていきますよー。頑張って付いてきてくださいね。

まず、printf()についての説明です。右のProgram2.1を見てください。見たことない行がありますね。

printf("Hello, world!!");

これは、「Hello, world」という文字列を出力しろ、という命令になります。実際に、Result2.1の実行結果を見てもそうだと分かります。

```
#include <stdio.h>

int main(void){
    printf("Hello, world!!");
    return 0;
}
```

▲ Program2.1　hello.c

```
Hello, world!!
```

▲ Result2.1　hello.cの実行結果

#2-1）printf()

　要するに、printf()の括弧の中に、表示したい文字列を書くだけです。簡単ですね。以下にprintf()の使い方を示します。

printf(文字列)

　気をつけなければならない点が2つあります。

1. C言語では""(ダブルクォーテーション・マーク)で括らないと、文字列として見なしてくれません。
2. C言語では全ての指示の末尾には;(セミコロン)をつけなければなりません。

間違ったプログラムを書くと、
コンパイル時にエラーが発生する!!

```
#include <stdio.h>

int main(void){
    printf(Hello, world!!);
    return 0;
}
```

▲ Program2.2　ダメな例(その1)

```
#include <stdio.h>

int main(void){
    printf("Hello, world!!")
    return 0;
}
```

▲ Program2.3　ダメな例(その2)

#2-1）printf()

　さて、printf()の使い方をもう少し拡張してみましょう。右のProgram2.4を見てください。

　いきなりaとかbとかcとかが出てきましたね。これらは変数と呼ばれるものです。この際ですから、変数についても少し勉強しておきましょう。

　まず、変数を使うためには宣言が必要です。「int a, b, c;」といった行がこれにあたります。ここでは、「これからint型※の変数であるaとbとcを使いますよ」と予告しています。ここで宣言していない名前の変数は使うことができないので注意してください。なお、変数の宣言も指示の1つなので、末尾に;を忘れずに。

```
#include <stdio.h>

int main(void){
    int a, b, c;
    a = 3;
    b = 7;
    c = a+b;
    printf("aとbの和は%dです。", c);
    return 0;
}
```

▲ Program2.4　add.c

※次回以降詳しく取り扱いますが、int型とは整数型のことです。

#2-1）printf()

　宣言しただけでは、いわば名前の付いた空箱ができただけなので、実際に箱の中に数字を入れていきます。

　=(イコール)は代入演算子と言います。「a = 3」でaに3を代入し、「b = 7」でbに7を代入、最後に「c = a+b」でaとbとの和をcに代入したことになります。

　このとき、当然のことですが、cの中身は10になっています。

```
#include <stdio.h>

int main(void){
    int a, b, c;
    a = 3;
    b = 7;
    c = a+b;
    printf(" aとbの和は%dです。", c);
    return 0;
}
```

▲ Program2.4　add.c

#2-1）printf()

さて、問題のprintf()部分ですね。先ほどと若干書式が異なります。

printf("aとbの和は%dです。", c);

文字列中に%dとありますね。これは変換指定文字と呼ばれるもので、引数を10進数として出力するものです。引数というのはカンマで区切った先にあるcのことになります。

つまり、この文字列中の%dは、cを10進数で表記したものに置き換えられるのです。

```
#include <stdio.h>

int main(void){
    int a, b, c;
    a = 3;
    b = 7;
    c = a+b;
    printf(" aとbの和は%dです。", c);
    return 0;
}
```

▲ Program2.4　add.c

#2-1）printf()

cの中身は10だったので、10進数表記にしても10ですね。したがって、実行結果は右のResult2.4のようになります。

このように、引数をとるときのprintf()の使い方は、以下のようになります。

printf(変換指定文字を含む文字列, 引数)

```
#include <stdio.h>

int main(void){
    int a, b, c;
    a = 3;
    b = 7;
    c = a+b;
    printf(" aとbの和は%dです。", c);
    return 0;
}
```

▲ Program2.4　add.c

```
aとbの和は10です。
```

▲ Result2.4　add.cの実行結果

#2-1）printf()

ちなみに、printf()において、引数の個数は1つである必要は全くありません。

右のProgram2.5に示した通り、カンマで区切りさえすれば、いくつでも引数をとることができます。

ただし、変換指定文字と引数は1対1で対応しているので、変換指定文字と引数の個数は同じになるようにしなければなりません。

```c
#include <stdio.h>

int main(void){
    int a, b, c;
    a = 3;
    b = 7;
    c = a+b;
    printf("%dと%dの和は%dです。", a, b, c);
    return 0;
}
```

▲ Program2.5　add2.c

```
3と7の和は10です。
```

▲ Result2.5　add2.cの実行結果

Column）改行したい!!

皆さん、ふと改行したくなる時ってありませんか。ありますよね。あるんだよ!!

ということで、実行結果における改行の仕方を教えます。改行をするためには、文字列の改行したい部分に¥nと書いておく必要があります。実際に使用した例が、右のProgram2.5になります。

改行をすると実行結果が見やすくなるので、printf()中の文字列の末尾には、¥nと書いておく癖をつけた方が良いですよ。

```c
#include <stdio.h>

int main(void){
	int a, b, c;
	a = 3;
	b = 7;
	c = a+b;
	printf("aは%dです。¥n", a);
	printf("bは%dです。¥n", b);
	printf("aとbの和は%dです。¥n", c);
	return 0;
}
```

▲ Program2.6　add3.c

※環境によっては、「¥」がバックスラッシュとして表示されます。

Column）改行したい!!

```c
#include <stdio.h>

int main(void){
	int a, b, c;
	a = 3;
	b = 7;
	c = a+b;
	printf("aは%dです。¥n", a);
	printf("bは%dです。¥n", b);
	printf("aとbの和は%dです。¥n", c);
	return 0;
}
```

▲ Program2.6　add3.c

```
aは3です。
bは7です。
aとbの和は10です。
```

▲ Result2.6　add3.cの実行結果

Column）改行したい!!

```
#include <stdio.h>

int main(void){
	int a, b, c;
	a = 3;
	b = 7;
	c = a+b;
	printf("aは%dです。", a);
	printf("bは%dです。", b);
	printf("aとbの和は%dです。", c);
	return 0;
}
```

▲ Program2.7　add4.c

```
aは3です。bは7です。aとbの和は10です。
```

▲ Result2.7　add4.cの実行結果

¥nを書いていないので、
実行結果が改行されずに
1行で出力される。

#2-2）scanf()

さて、scanf()の話に移りましょう。printf()が出力を司る命令なのに対し、scanf()は入力を司る命令になります。

右のProgram2.8を見てください。次のような行がありますね。

```
scanf("%d", &h);
```

この行は、おおざっぱに言うと「今から10進数のデータをキーボードから入力するから、それを変数hにぶち込め」といった指示です。

```
#include <stdio.h>

int main(void){
    int h;
    printf("あなたの身長は何cmですか?¥n");
    printf("整数で入力してください。¥n");
    scanf("%d", &h);
    printf("あなたの身長は%dcmです。¥n", h);
    return 0;
}
```

▲ Program2.8　height.c

#2-2）scanf()

とりあえずプログラムを実行させてみましょう。

実行させると、まずResult2.8の上の状態になると思います。これはプログラムが終了してしまった訳ではなく、scanf()の入力待ちの状態です。

ここで、何らかの整数をキーボードから入力し、Enterを押します。すると、変数hにその数字が代入されて、そのまま最後までプログラムが進行していきます。

```
あなたの身長は何cmですか?
整数で入力してください。
```

```
あなたの身長は何cmですか?
整数で入力してください。
174 [Enter]
あなたの身長は174cmです。
```

▲ Result2.8　height.cの実行結果

#2-2）scanf()

scanf()の使い方をまとめると、以下のようになります。

scanf(変換指定文字を含む文字列, &変数名)

忘れてはならないのが、変数名の前に付ける&です。この&を忘れる人が毎年大量にいるようなので、皆さんは気をつけましょう……。

忘れてほしくないので、右にでっかく書いておきますね。

#2-2）scanf()

scanf()においても、1度に複数の値を入力することが可能です。

右のProgram2.9に示したようにscanf()部分を変えれば、いくつでも同時入力を受け付けることができます。

ただし、実際にキーボードから数字を同時入力する際には、数字同士をスペースで区切ることを忘れずに。スペースで区切らないと、1つの数字になっちゃいますからね。

今回の講義は以上です。お疲れ様!!

```c
#include <stdio.h>

int main(void){
    int h, w;
    printf("身長と体重を入力してね!!¥n");
    scanf("%d%d", &h, &w);
    printf("身長: %dcm　体重: %dkg¥n", h, w);
    return 0;
}
```

▲ Program2.9　height_weight.c

```
身長と体重を入力してね!!
174 68 [Enter]
身長: 174cm　体重: 68kg
```

▲ Result2.9　height_weight.cの実行結果

Question2-1）printf()

以下の(1), (2)のプログラムを実行したときの、実行結果をそれぞれ記せ。

(1)

```
#include <stdio.h>

int main(void){
        printf("Welcome to¥nWaseda");
        printf("University!!¥n");
        return 0;
}
```

▲ Program2.10　welcome.c

(2)

```
#include <stdio.h>

int main(void){
        int a, b, c;
        a = 1;
        b = 10;
        c = 100;
        printf("%d¥n", b);
        printf("%d¥n", a+100);
        printf("%d, %d¥n", a+b+c, 333);
        return 0;
}
```

▲ Program2.11　calc.c

Answer2-1）printf()

(1)

```
Welcome to
WasedaUniversity!!
```

▲ Result2.10　welcome.cの実行結果

printf()では、「¥n」があるところでのみ、改行が発生する点に注意しましょう。

(2)

```
10
101
111, 333
```

▲ Result2.11　calc.cの実行結果

引数の部分は、何も変数だけじゃありません。このように、計算式を引数として入れたり、数字を直接引数にすることも可能です。

Question2-2）scanf()

英語と数学の合計点を求めるプログラムである、Program2.12を作成した。

実行結果がResult2.12のようになるとき、Program2.12の空欄を正しく埋めよ。

```
#include <stdio.h>

int main(void){
    int x, y;
    printf("英語の点数は何点ですか?¥n");
    
    printf("数学の点数は何点ですか?¥n");
    
    
    return 0;
}
```

▲ Program2.12　total.c

```
英語の点数は何点ですか?
78 [Enter]
数学の点数は何点ですか?
86 [Enter]
合計点は164点です。
```

▲ Result2.12　total.cの実行結果

Answer2-2）scanf()

```
#include <stdio.h>

int main(void){
    int x, y;
    printf("英語の点数は何点ですか?¥n");
    scanf("%d", &x);
    printf("数学の点数は何点ですか?¥n");
    scanf("%d", &y);
    printf("合計点は%d点です。", x+y);
    return 0;
}
```

▲ Program2.12　total.c

左の解答は一例です。

自分の答えが合っているのかどうか分からないときは、実際にプログラムを書いて実行したり、近くの先輩に聞いたりしてみることをオススメします。

Question2-3）デバッグ

以下のProgram2.13には、致命的な欠陥が数カ所ある。それらを指摘し、訂正せよ。

```
#include (stdio.h)

int main(void){
    int age;
    printf(あなたの年齢は何歳ですか?¥n);
    scanf("%d", age);
    printf(あなたは10年後には、%d歳です。¥n, Age+10);
    return 0
}
```

▲ Program2.13　age.c

Answer2-3）デバッグ

```
#include <stdio.h>      ← ()ではなく〈〉

int main(void){
    int age;
    printf("あなたの年齢は何歳ですか?¥n");     ← ""抜け
    scanf("%d", &age);      ← &抜け
    printf("あなたは10年後には、%d歳です。¥n", age+10);     ← ""抜け、変数名が違う
    return 0;     ← ;抜け
}
```

このように、メチャクチャ欠陥だらけでした。全て指摘できましたか？

プログラムにおいては、大文字小文字を1文字間違えただけで動かなくなるなんてことはザラにあります。気をつけてくださいね。