目 次

第1章　第２回放送

1.1　第２回放送内容

1. 変数
2. 演算
3. キャスト
4. scanf 文
5. 計算練習

1.2　変数

1.2.1　変数とは

何か計算をしたとき、その計算結果を覚えておきたいときがあると思います。その時使うものが変数です。変数とは、値を保存することができる箱です。

1.2.2　代入・型

値を収納できる箱を作ったら、つぎはその箱の中に値を入れてみましょう。この箱に何かを入れるという作業を代入といいます。しかし、箱にはどんな値（物）でも入れることはできません。日常に結びつけにくいのですが、例えばパソコンを入れる専用の大きな箱とお菓子を入れる専用の小さな箱があるとします。パソコン専用の箱にはパソコンを入れることができ、同様にお菓子を入れる専用の箱にはお菓子を入れることができます。ここでお菓子を入れる箱にパソコンを入れようとします。入らないことは無いのですが、パソコンは大きいので、お菓子を入れる箱にちょっとしか入りません。逆に、パソコンを入れる箱にお菓子を入れると、お菓子は小さいので、箱には余分なスペースがかなり残ってしまうでしょう。

このように、用途によって箱を使い分けますが、C 言語でも同様に用途によって変数（箱）を使い分けます。この箱の種類の事を型といいます。

1.2.3　型の種類

では、具体的にどのような型があるのでしょうか。C 言語では次のような型が存在します。

```
short 型　-32768 〜　+32767
int 型　　-2147483648 〜　+2147483647
long 型　 -2147483648 〜　+2147483647
unsigned short 型　0〜+65535
unsigned int 型 0 〜 +4294967295
unsigned long 型 0 〜 +4294967295
char 型　-128〜+127
unsigned char 型 0〜+255
float 型 3.4 * 10^-38 〜　3.4 * 10^+38
double 型 1.7 * 10^-308 〜　1.7 * 10^+308
```

整数を扱う場合は int 型を、少数を扱う場合は double 型を、文字を扱う場合は char 型を使う場合が多いです。[1]

1.2.4　変数の作り方

では、実際変数を作ってみましょう。int 型の変数を作りたい場合、次のようにします。

```
int hoge;
```

これは、hoge という名前の int 型の箱を作れという意味です。箱には名前をつけることができます。同様に double 型の piyo という名前の箱を作りたい場合は、

```
double piyo;
```

とします。

[1] これは、処理系によって若干違う場合があります。

1.2.5 変数の名前のつけ方の注意

変数名を付ける際に、次のようなルールがあります。

- 先頭は英字か_(アンダースコア）でなければならない
  数字ではダメです。
- 同じ名前の変数が２つ存在する
- 予約語は使えない
  予約語とは、int や double などです。

1.2.6 値の代入

では、箱に何か値を入れてみましょう。例えば、10 という値を箱に入れたい場合は次のようにします。

```
int hoge;
hoge = 10;
```

この = という記号は、式の右側の値を左側に代入せよ、という意味です。数学のイコールという意味ではないので注意してください。では、数学でよくある x = 10（x は１０である）といった本当にイコールの意味を使いたい場合はどうすればよいでしょうか。これは、if 文という所でしっかり説明しますが、今軽く説明しておくと、== という記号を使います。イコールを二つ繋げて書くと、本当に数学でいうイコールという意味になります。

話を戻して、ここで箱を作った時点で値を入れた状態にしたい場合（これを初期化という）は、`int hoge = 10;` とします。

1.2.7 printf 文再び

では、変数を画面上に出力したい場合、どのようにしたらよいでしょうか。例として、int 型の hoge という箱に１０という値を代入して、その箱の中身を画面上に表示させてみます。

```
int hoge;
hoge = 10;
printf("%d",hoge);
```

%d の部分は、続く実引数を整数の１０進法で表示せよ、という意味です。実引数とは、「,」の後に続く部分のことをいいます。このプログラムを実行すると、次のように画面上に表示されると思います。

実行結果

```
10
```

１０進法で表示があるのだから、８進法で表示や１６進法で表示というものも当然存在します。また、小数点などを表示したい場合（double 型や float 型の場合）には、別の記号を使います。

%d 整数の１０進法として出力
%u 符号なし整数の１０進法として出力
%o 整数の８進法として出力
%x 整数の１６進法として出力
%f 小数点表示（double や float の場合に使う）
%c １文字出力
%s 文字列を出力
%p ポインタの値を出力
%% %を出力

この%で始まる部分を変換指定子といいます。
他にもいろいろ試して見ましょう。

```
int hoge = 10;
int piyo = 20;
double foo = 30.256;

printf("hoge の値は%d です\n",hoge);
printf("piyo の値は%d です\n",piyo);
printf("値は%d です\n", 50);
printf("hoge は%d piyo は%d です\n",hoge,piyo);
printf("foo の値は%f です\n",foo);
printf("foo の値は%d です\n",foo);
```

1.2.8 変数の注意

```
int hoge;
printf("hoge の値は%d です",hoge);
```

としたらどうなるでしょうか。出力結果はわかりません。もしかしたら 0 とでるかもしれませんし、分けのわからない数字が出力されるかもしれません。このように、変数を作った時点では中にはゴミが入っています。そのため、変数は必ず何か値を代入してから使いましょう。

1.3 演算

1.3.1 四則演算

数学でもお馴染みの四則演算、つまり「足し算」「引き算」「掛け算」「割り算」をC言語でも行うことができます。その方法は簡単です

記号

＋ 足し算

- 引き算

* 掛け算

/ 割り算

1.3.2　簡単な計算の例

```
int x, y;

x = 10;
y = 20;
printf("x+y は%d です\n",x+y);
y = y - x;
printf("y の値は%d です\n"y);
```

1.3.3　商を求める際の注意

商、つまり割り算を求めるときに注意すべき点があります。たとえば以下の場合です。

```
int x = 7;
int y = 3;

printf("x/y の値は%d です",x/y);
```

この計算では、小数点以下を切り捨てた値が出てきます。なぜなら、int 型は整数しか箱に値を入れることができないからです。こういう場合は

```
double x = 7;
double y = 3;

printf("x/y の値は%f です",x/y);
```

とすれば問題ないです。また、y の値を 0 にして、0 で何か値を割ってはいけません。

1.3.4　その他の演算

値の符号の反転をしたい場合は次のようにします。

```
int x = 10;
printf("反転した値は%d です\n",-x);
```

剰余、つまり割り算のあまりを求めたい場合は、次のようにします。

```
int x = 10, y = 3;
printf("余りは%d です\n",x % y);
```

1.3.5 略した式

```
int x = 10 , y = 20;
x += 10;
```

この x+= 10 というのは x = x + 10 を略した書き方です。これらの計算はよく使われるので、こうした略が存在します。他の演算も同様に

```
x -= 10;
x /= 10;
```

などが存在します。
また、x に 1 を足したい場合があるとします。x += 1;
この 1 を足すという作業は、特によく使われるので、更に略して

```
x++;
```

という書き方をしても良い事になっています。この書き方をインクリメントといいます。同様に、デクリメント、つまり値を 1 引くという書き方も存在します。

```
x--;
```

1.3.6 べき乗の計算の注意

べき乗とは、例えば x の 2 乗、つまり x*x などのことをいいます。よく、x^2 という表現が使われますが、C 言語ではこれでは 2 乗という意味になりません。^は xor（排他的論理輪）という意味になってしまいますが、今はあまり気にしなくてよいです。2 乗の計算をしたい場合は、素直に x*x としてください。ここで、3 乗、4 乗 … と増えていくと、x*x*x*x* … となってしまいます。これらを回避する方法として、pow という関数が存在しますが、これも今は気にしないでください。簡単に説明しますと、pow は次のように使います。

```
pow(x,4.0);
```

これは x の 4 乗という意味です。

1.4　キャスト

1.4.1　違う型どうしの計算

例えば次のようなコードを書いたとします。

```
double dx = 10, dz;
int iy = 3;

dz = dx / iy;
```

このようにintとdouble型の計算をした場合は、格上げという暗黙の型変換が行われています。ここでは、iyは型が勝手にdouble型に変換されます。暗黙の型変換ではより大きな型に合わせて変換されます。たとえば、intとdoubleでは、intは整数だけを収納できるのに対し、doubleは整数も収納できるし、少数も収納することができます。よってこの場合は、intはdoubleに型変換されることになります。

1.4.2　キャスト

暗黙の型変換ではなく、明確に型変換をしたい場合次のようにします。

```
double dy = 10, dz;
int iy;

dz = (int)dy / iy;
```

このようにdyをint型に変えたい場合、dyの前に(int)と書きます。同様にdouble型にしたい場合は、(double)とします。こういった型変換をキャストと呼びます。

1.5　scanf文

1.5.1　キーボードからの読み込み

キーボードから何か値を入力して、変数の中に代入したい場合があると思います。そういった場合に使うのがscanfです。次のソースをコンパイルして実行すると、キーボードから読み込んだ文字を画面上に出力します。

```
#include <stdio.h>

int main(void)
{
int x;

printf("数字を入力してください:");
scanf("%d", &x );
printf("キーボードから %d という数字を受け取りました\n" , x);
return 0;
}
```

scanf は printf と書き方が似ていますが、x の前に&が付いていることに注意してください。この&の意味はポインタのところで詳しく説明しますので、今は&をつけなければならないという事を忘れないでください。scanf 文の中に書いてある%d は printf の時と同様、１０進法の値をキーボードから読み込んで、その値を x に収納せよということです。ただし、double 型や float 型を読み込みたい場合、%f ではなく%lf となることに注意してください。

1.6 計算練習

今日学んだ事のまとめとして、いろいろ計算をしてみたいと思います。

1.6.1 分を時間に直す

例えば、180 分と入力したら 3 時間と表示するプログラムを作ってみたいと思います。

```
#include <stdio.h>

int main(void)
{
double hour,minute;

printf("何分かを入力してください:");
scanf("%lf",&minute);

hour = minute / 60.0;
printf("結果は%f 時間です\n",hour);

return 0;
}
```

他にいろいろ計算してみてください。

1.7　本日の講義内容のおさらい

- 変数とは

  値を収納できる箱である。箱は用途に合わせて使用する。

- 演算

  四則演算、インクリメントなどの略した式が存在する。

- キャスト

  型の違うもの同士で計算する時に使う

- scanf 文

  キーボードから文字を受け取る