目 次

第1章　第5回放送

1.1　第5回放送内容

1. 文字コード
2. 関数

1.2　文字コード

1.2.1　文字コードについて

C 言語には文字コードというものが存在します。文字コードとはなんのことでしょうか。ずいぶん前の放送で 1 文字を保存するために使う変数は char 型だと説明しました。そして char 型の中に入れることができる値は整数の -128 〜 127 だと言いました。つまり

```
char hoge;
hoge = 97;
```

は問題ないです。これは hoge に 97 という値を代入するということです。しかし、char 型は普通次のように使っていました。

```
char hoge;
hoge = 'a';
```

さっき char 型に収納できる値は整数だといいましたが、ここでは文字を代入しているように見えますね。実はこれら 2 つのコードは全く同じ意味です。一番最初の講義に説明したとおり、コンピュータは 1 と 0 しか理解することができません。この 1 と 0 を組み合わせると整数を表現することができます。例えば、10 進法で

10 を 2 進法に直すと 1010 となります。しかし 1 と 0 ではいくら頑張っても文字を表現することができません。ここで登場するのが文字コードとなります。つまり、文字コードとは単なる整数を決められた文字に置き換える役割を果たします。

1.2.2　文字コード表

では、文字コードとはどのようになっているのでしょうか。次に表す表 (図 1.1) は日本で一般的に使われている JIS コード表です。

図 1.1: JIS コード表

文字コード表では 16 進法で表されています。この文字コード表より、'a' は 16 進法で 61、10 進法で 97 となります。この '（シングルクウォーテーション）で囲まれた 1 文字が文字コード表により整数に置き換えられます。

1.2.3 16 進法の使い方

値を 16 進法で代入する場合、数字の前に 0x をつけます。

```
char hoge = 0x61;
```

1.2.4 数字の注意点

文字コード表を良く眺めてください。文字コードには数字が含まれています。例えば、

```
char hoge = '1';
```

と書いたとしましょう。これは、

```
char hoge = 1;
```

とは違います。'（シングルクォーテーション）で囲むと文字コード表の整数に置き換えられるという点に注意してください。つまり'1' は 0x31 になります。

1.2.5 日本語の話

文字コード表を見ると全部で 256 通りの文字しか表現することができません。しかし日本語は漢字、ひらがな、カタカナとどう考えても 256 文字では足りません。日本語の場合はとても特殊で 2 つの char 型で一文字を表現します。これらの日本語の話はとても複雑なので省略します。

1.2.6 文字化けの話

文字コードは 1 種類ではありません。文字コードにはいろいろな種類が存在します。これらの文字コードの配置の違いにより、適切な文字コードを選択しないと文字化けという現象が起こります。

1.3　関数

1.3.1　関数とは

C 言語に欠かせないものの一つとして関数というものが存在します。実は今までの放送で関数を何回も使っていました。例えば printf ですが、これは関数です。scanf も関数です。関数とはあるまとまった処理をする 1 つの機能です。

1.3.2　関数を作ってみる

2 つの値を入力して、その和を求める関数を作ってみましょう。

```
#include <stdio.h>

int Plus(int x, int y)
{
int result;
result = x + y;
return result;
}

int main(void)
{
int hoge,piyo;
int n;

printf("2 つの整数を入力してください\n");
printf("1 つ目:"); scanf("%d",&hoge);
printf("2 つ目:");  scanf("%d",&piyo);

n = Plus(hoge,piyo);
printf("足した結果は%d です\n",n);

return 0;
}
```

さて、少しずつ解説していきましょう。まずC言語における関数の書き方は次のようになります。

関数の書き方

```
型 関数名（引数）{ 関数の中身 }
```

まず注意して欲しいのは、main 文の前に Plus 関数を書いてください。これは後でプロトタイプ宣言というところで説明します。

関数名に自分が作った関数に名前をつけてください。ただし、すでに使われている関数名や変数名は使わないでください。この場合は、printf などの名前をつけないでください。

関数名の前に型とあります。これはその関数が値を返すときの型となります。例えば、Plus 関数の場合、main 文の中で `n = Plus( hoge , piyo);` となっています。これは Plus 関数で処理した内容を n の中に代入する際に、n の型は関数の型と一致していなければなりません。つまり n は int 型でなければなりません。

さて、関数を呼び出す際に `Plus( hoge, piyo );` としました。関数を呼び出す際には、 関数名 (引数) という風にします。引数の説明は図 1.2 を使って説明します。

図 1.2 は引数の様子を示した図です。呼び出した側の引数を実引数といい、呼び出された側の引数を仮引数といいます。このとき、実引数の値が仮引数の値にコピーされます。つまり、

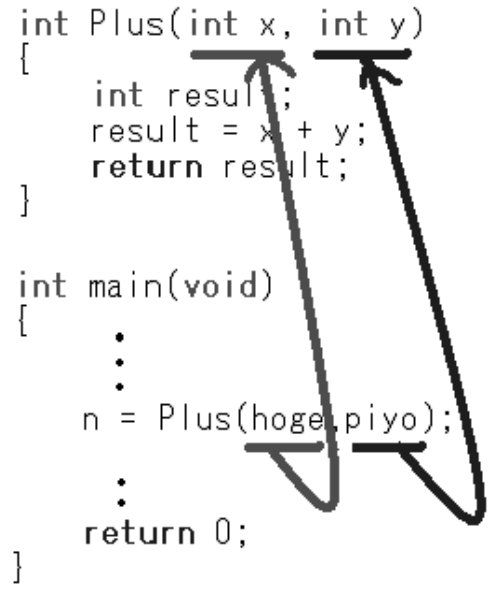

図 1.2: 引数

```
void Test(int x)
{
   x = 10;
}

int main(void)
{
  int hoge;
  Test( hoge )

  return 0;
}
```

としても x は hoge のコピーなので、x の中身をいくら変更しても hoge には影響を及ぼしません。また、Test 関数ですが、値を返す必要がないときは関数の型を void とします。引数に関してですが、もしかしたら引数が何も必要が無い場合が

あるかもしれません。そういった場合は、引数の中に void、つまり空という宣言を書きます。

```
void Test2(void)
{
printf("TEST FUNCTION\n");
}

int main(void)
{
Test2();
        return 0;
}
```

最後に return ですが、Plus 関数の例では result という変数の中身の値を返せという意味です。関数は return を見つけるとその場でその関数の役割は終了します。

1.3.3 プロトタイプ宣言

C 言語で書いたコードは上から下へ実行されます。では、例として次のようなコードを書いたとします。

```
#include <stdio.h>

int main(void)
{
Test2();
        return 0;
}

void Test2(void)
{
printf("TEST FUNCTION\n");
}
```

このコードをコンパイルするとエラーが発生するでしょう。なぜなら、main 文の中で Test2 関数を呼び出しているが、この時点ではまだ Test2 関数が本当に存在しているかどうかを判断することができません。このような場合に使われるのが、プロトタイプ宣言とよばれるものです。

```
#include <stdio.h>

void Test2(void); //プロトタイプ宣言

int main(void)
{
Test2();
        return 0;
}

void Test2(void)
{
printf("TEST FUNCTION\n");
}
```

プロトタイプ宣言を行ってみました。こんどは main 文より前にあらかじめ Test2 という関数が存在しますよ、ということを知らせるためにこのような書き方をします。Test2() のあとにあるセミコロン (;) に注意してください。

1.3.4　main 関数

私たちが一番よく使っている関数に main 関数があります。main 関数は私たちがコードを書く際に一番初めに実行される関数です。と書いておきながら、本当は一番初めではありません。実は main 関数もどこか別のところから呼び出されています。メイン関数が呼び出されている所をみてみましょう。[1]

```
int BP_SYM( ··· )
{
```

[1] 実際のコードではありません。

```
...

result = main(argc,argv,__environ);

...

exit(result);

}
```

こんな感じに呼び出されています。これをみてわかるとおり、main は値を返してその値の結果を元に exit という関数を呼び出して処理を終了しています。exit 関数の引数に 0 を渡すと正常終了し、0 以外の値を渡すと異常終了を表します。このことより、main 文で異常が発生した場合、

```
int main(void)
{
return 1;
}
```

などとすればよいです。また、よくある間違いですが、main 関数を作る際には

```
void main(void)
{
//何か処理
}
```

では駄目です。理由はいままでの説明を読んでいただければわかると思います。

1.4　演習問題

1.4.1　文字コード表を表示するプログラム

仕様

０～２５５までの文字コードを表示するプログラムを作成せよ。表示結果は10進法の値、16進法の値、文字を表示せよ。

```
0 0
1 1

(略)

65 41 A
66 42 B

（略）
```

```
#include <stdio.h>

int main(void)
{
int hoge;

        for( hoge = 0; hoge < 256 ; hoge++ ) {
         printf("%3d %3x %c\n",hoge,hoge,hoge);
        }

        return 0;
}
```

1.4.2 べき乗を求めるプログラム

仕様

べき乗を求めるプログラムを作成せよ。例えば2の５乗を求める関数を作る場合は、Power(2 , 5) となるようにすること。

```
#include <stdio.h>

int Power(int x, int y); //プロトタイプ宣言

int main(void)
{
int xi,yi;

        printf("整数を入力してください:");
        scanf("%d",&xi);
        printf("何乗にしますか:");
        scanf("%d",&yi);

        printf("%d の%d 乗は%d です",xi,yi,Power(xi,yi));
        return 0;
}

int Power(int x, int y)
{
int i,result;

        for( i=1, result = 1; i <= y; i++) {
         result *= x;
        }
        return result;
}
```

1.5　本日の講義のおさらい

- 文字コード
  数字を文字に置き換えたもの
- 関数
  あるまとまった処理をするもの

参考文献

[1] Brian W. Kernighan, Dennis M. Ritchie: The C Programming Language Second Edition.

[2] ハーバートシルト著 トップスタジオ訳: 独習 C 第三版.

[3] 田中 敏幸著 C 言語によるプログラミングの基礎

[4] 柴田 望洋 明解C言語

[5] 松本 健一 上田 悦子 安室 喜弘 ２００６年度プログラミング演習（初級コース）課題

http://chihara.naist.jp/people/STAFF/yasumuro/Pub/c-ensyu2006/

[6] Ryo Kawahara C/C++プログラミング初心者講座

http://www.stat.phys.kyushu-u.ac.jp/~ryokawa/cbegin2_3/pdf/cbegin.pdf

[7] TOMOJI 初心者のためのポイント学習 C 言語

http://www9.plala.or.jp/sgwr-t/

[8] 小出 俊夫 C言語入門インターネット版

http://homepage1.nifty.com/toshio-k/prog/c/

[9] 赤坂 玲音 C 言語入門

http://wisdom.sakura.ne.jp/programming/c/index.html