ソフトゼミ A 2016 第 7 回

ポインタ

今回でゼミ A は最終回です。今までお疲れ様でした。

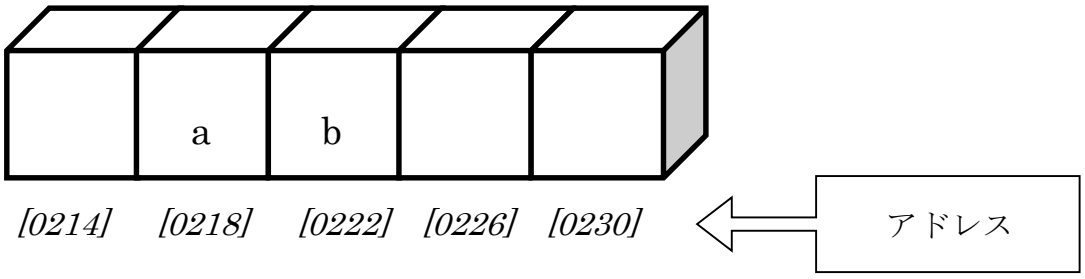
さて、今回解説するのは「ポインタ」についてです。C 言語の中ではちょっと難しい概念とも言われますが、一つ一つしっかりと理解していきましょう。わからないことがあれば遠慮せずに質問してください。

➢ アドレス

ポインタとは「変数のアドレスを保存できる特別な変数」のことです。まずはアドレスについて説明します。

変数が宣言されると、そのデータを保存しておくためにコンピュータのメモリの一部分が使われます。このとき、変数のメモリ上での「場所」を表しているのがアドレスです。

例えば、「int a,b;」のように int 型の変数 a,b を宣言すると



【メモリのイメージ】

上の図のように、メモリ上の適当な場所に各変数が割り当てられます。

変数のアドレスは &変数名 で表すことができます。上の図で説明すると、a のアドレスは &a で表すことができ、その値は *0218* となります。scanf 文で出てきた「&」も同じ意味です。

➢ ポインタ

それでは、ポインタの説明に移ります。ポインタとは「変数のアドレスを保存できる特別な変数のこと」であり、先ほどの図における「0214」、「0218」、「0222」…というような値を保存できるというわけです。

続いてポインタの使い方を説明していきます。

➢ 宣言

まずはポインタの宣言です。宣言の仕方は以下の通りです。

```
データの型　*変数名(ポインタ名) ;
```

具体的には「int *a;」のようにします。この時、*(アスタリスク)は「宣言した変数がポインタである」という目印であって、変数の名前には含まれないので注意です。また「データの型」を int とすれば、int 型の変数のアドレスが保存できるポインタが宣言され、同様に char とすれば char 型へのポインタが宣言されます。

ちなみに「int 型の変数のアドレス、を保存できる(されている)ポインタ」のことを「int 型へのポインタ」というふうに言います。

➢ 代入

ポインタにアドレスを代入するときには以下のようにします。

```
int a=20;      //int 型の変数の宣言
int *p;        //int 型へのポインタの宣言
p=&a;          //変数のアドレスをポインタに代入
```

こうして変数 a のアドレスがポインタ p に格納されます。1 ページ目の図でいえば、ポインタ p には 0218 が代入されたことになります。このように、ポインタ p に変数 a のアドレスが格納されているとき、「p は a を指している」といいます。

またポインタ p が変数 a を指しているとき、*p は「ポインタ p が指す変数」すなわち変数 a を表します。このときの * は宣言時の * とは別物と考えて差し支えありません。全くもって別物というわけではないのですが…

➢ 表示

ではこれまでのことを含めて、プログラムの例を見てみましょう。

```
#include<stdio.h>
int main(void){
    int a=20;
    int *p;
    p=&a;
    printf("a=%d¥n",a);                    //a の値を出力
    printf("a のアドレス:%p¥n",&a);    //a のアドレスを出力
    printf("p:%p¥n",p);      //p の値、すなわち a のアドレス
    printf("*p=%d¥n",*p);                   //*p は a と書くのと同じ
    return 0;
}
```

%p はアドレスを printf で出力するためのものです（int に対する%d と同じです)。

➢ 交換

前回のゼミ A の「グローバル変数」の項でふれましたが、本来、ローカル変数の値を外部の関数が書き換えることはできません。しかしポインタを使うと外部の関数からローカル変数を書き換えることが可能となります。

ではプログラムの例を見てみましょう。main 文で宣言された変数 a,b を、関数を使って交換するプログラムです。わかりやすくするために、ポインタを使わずに値を交換しようとする関数 swap0 もあります（実行すればわかりますが、値は交換されません)。

```
#include<stdio.h>
void swap0(int a,int b){              //ポインタを使わずに交換
    int c;
    c=a;      //a の値を、c に代入
    a=b;      //b の値を、a に代入
    b=c;      //c の値を、b に代入
}
void swap(int *a,int *b){             //ポインタを使って交換
    int c;
    c=*a;     //a が指す変数の値を、c に代入
    *a=*b;    //b が指す変数の値を、a が指す変数に代入
    *b=c;     //c の値を、b が指す変数に代入
}

int main(void){
    int a=20,b=30 ;
    printf("a=%d b=%d¥n",a,b);
    swap0(a,b);
    printf("a=%d b=%d¥n",a,b);
    swap(&a,&b);
    printf("a=%d b=%d¥n",a,b);
    return 0;
}
```

実行結果は次のようになるはずです。

```
a=20 b=30
a=20 b=30
a=30 b=20
```

コンパイル時に警告が出ると思いますが、無視して大丈夫です。

それでは順に説明していきましょう。

まず実行結果の１行目ですが、a,b の初期値をそのまま表示しているだけです。説明は不要だと思います。

実行結果 2 行目では、関数 swap0 で値を交換した結果が表示されています。が、値の交換ができていないことは一目瞭然です。交換ができていない理由は、swap0 で交換している a,b は swap0 のローカル変数であり、main のローカル変数である a,b ではないからです。

実行結果 3 行目では、関数 swap で値を交換した結果が表示されています。こちらはちゃんと値が交換できています。なぜならば、swap で交換しているのは swap のローカル変数(ローカルポインタ)a,b が指す変数、すなわち main のローカル変数 a,b だからです。つまり、外部から変数の値を直接書き換えているわけです。

このようにポインタをうまく使えば、関数外からローカル変数を直接操作することができるのです。

➢ 配列とポインタ

ほかの変数と違って、配列は配列名を関数に渡すことで、ローカルな配列であっても、外部から値を操作することが可能です。具体例を見てみましょう。

```
#include<stdio.h>
void array(int a[],int n){
    int i;
    for(i=0;i<n;i++){
        a[i]=i;
    }
}
int main(void){
    int a[10],i;
    array(a,10);
    for(i=0;i<10;i++){
        printf("a[%d]=%d¥n",i,a[i]);
    }
    return 0;
}
```

関数 array は配列名と要素数を受け取って、配列にその添え字と同じ数値を代入する関数です。
実行するとわかりますが、ちゃんと配列の値が書き換えられています。配列名を関数に渡すことは、実質的に配列そのものを渡したことと同じです。

第 5 回のゼミ A「構造体」の文字列(string)の項で、「配列名とポインタには切っても切れない関係がある」とチラッと言いましたが、実は配列名はその配列の先頭要素へのポインタなのです。(つまり a と&a[0]は同じです。)
そして、a[n] というのは「ポインタ a が指す変数から n 個隣にある変数」という意味になります。
そのため、先ほどの例のように関数外からでも値の書き換えができるというわけです。

➢ 練習問題

1. 配列を宣言し、その配列の各要素のアドレスを表示するプログラムを作成してください。関数の作成は任意です。

2. 3 つの変数を引数として受け取って、それぞれの変数に適当な値を scanf 文で入力する関数を作成してください。

以上でソフトゼミ A の内容は終了です。お疲れ様でした！
わからなかったことがあれば、サークル員に聞いたり自分で調べたりしてください。

次のページからは、ソフトゼミ B に関するお知らせです。

➢ ゼミ B について

今度はゼミ B が始まります。ゼミ A で学んだこと＋α を応用して簡単なゲームを作っていきます。乞うご期待。

日程は以下の通りです。

ソフトゼミB日程表			
種別	回数	活動日	内容
導入	1	5月26日	VC++とDXライブラリの導入
シューティング	2	5月31日	画像処理と移動
	3	6月2日	自機・敵機の弾の描画と移動
	4	6月7日	当たり判定
	5	6月9日	ファイル分割
アクション	6	6月14日	プレイヤーの移動と重力
	7	6月16日	ブロック判定
	8	6月21日	ファイル分割

ゼミ B 開始に伴い、導入等の操作が再び必要になります。余裕のある方はゼミ B 初回までに VC++と DX ライブラリを導入しておいてください。

■ Microsoft Visual Studio Community 2015 の導入

https://www.visualstudio.com/products/visual-studio-community-vs

上のページから「Visual Studio Community 2015」ダウンロードしてインストールしておいてください。

また、

http://www.meiji.ac.jp/isc/msca/

ここに書かれている通り、明大生は Visual Studio の有償版を無料で使うことができるみたいです。

インストールしたら Visual Studio を起動して「ヘルプ」→「製品の登録」から、ユーザー登録をしておいてください。

といっても Visual Studio Community は無償ですが。

■　DX ライブラリの導入

今回のゲーム制作にあたり、ゲーム制作を大幅に手助けしてくれるツールです。

http://dxlib.o.oo7.jp/index.html （←DX ライブラリのホームページ URL）

上のページの「DX ライブラリのダウンロード」の項目から、「DX ライブラリ Visual C++用」をダウンロード→解凍しておいてください。解凍したフォルダは C ドライブ直下などわかりやすい場所に置いておくことをお勧めします。

また、デスクトップに解凍したフォルダ「DxLib_VC/help」の中にある index.html のショートカットを作成しておくと、今後のゲーム制作で役立つと思います。（上に示した DX ライブラリのホームページです。）