プログラミング初級講義実況

第7回

第7回講義内容

- ポインタ入門

ポインタ

ポインタ入門

- T*型の変数は、T型のオブジェクトのアドレスを保持できる。

```
int* hoge;
  又は
int *hoge;
```

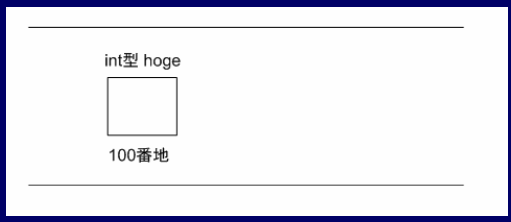

ポインタ宣言時の注意

```
int  *hoge, *piyo;
int*  hoge,  piyo;
```

後者はhogeがポインタ、piyoがint型になってしまっている。

値の代入

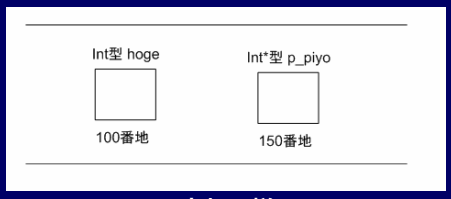
int hoge;



メモリ内部の様子

値の代入

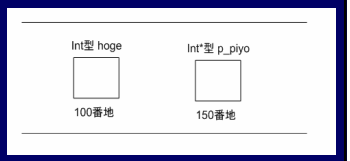
```
int hoge;
int* p_piyo;
```



メモリ内部の様子

値の代入

```
int hoge;
int* p_piyo;
hoge = 10;
p_piyo = &hoge;
```

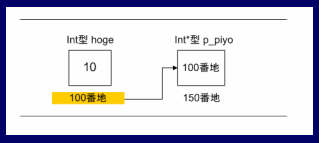


メモリ内部の様子

値の代入

```
int hoge;
int* p_piyo;
hoge = 10;
p_piyo = &hoge;
```

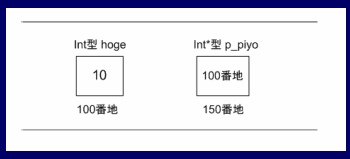
&をつけると、その変数の住所を取得せよ、という意味になる



メモリ内部の様子

変数の中身

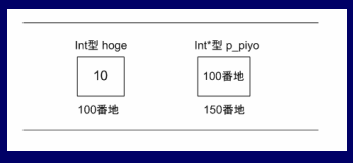
```
int hoge;
int* p_piyo;

hoge = 10;
p_piyo = &hoge;
```




```
printf("hoge address :%p¥n",&hoge);
printf("hoge :%d¥n",hoge);
printf("p_piyo :%p¥n",p_piyo);
printf("*p_piyo:%d¥n",*p_piyo);
printf("p_piyo address :%p¥n",&p_piyo);
```


変数の中身

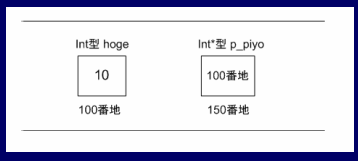
```
int hoge;
int* p_piyo;

hoge = 10;
p_piyo = &hoge;
```

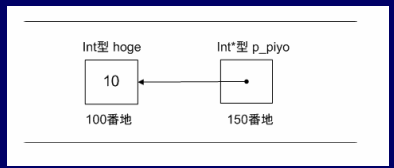



```
printf("hoge address :%p¥n",&hoge);
printf("hoge :%d¥n",hoge);
printf("p_piyo :%p¥n",p_piyo);
printf("*p_piyo:%d¥n",*p_piyo);
printf("p_piyo address :%p¥n",&p_piyo);
```

*p_piyoはp_piyoが持っているアドレスにあるデータを持ってくる

ポインタという名前





NULLポインタ

int hoge;
printf(“%d”,hoge);

結果はどうなるか？

NULLポインタ

int* p;
p = NULL;

NULLポインタ

#define NULL 0
#define NULL((void*)0)
#define NULL -1

NULLポインタ

int* p;
p = NULL;
printf(“%d”,*p);

変数の型の大きさ

- short**型** -32768~+32767
- int **型** -2147483648~+2147483647
- long **型** -2147483648~+2147483647
- char **型** -128 ~ +127 (unsigned)
- float **型** 3.4*10^-38 ~ 3.4*10^+38
- double**型** 1.7*10^-308 ~ 1.7*10^+308

変数の型の大きさ

- char**型は**1**バイト**
- int**型は**4**バイト**
- **（・・・の場合が多い）**

sizeof

```
#include <stdio.h>
int main(void)
{
	char hoge;
	int piyo;

	printf("char型のサイズは%uです", (unsigned int)sizeof(hoge));
	printf("int型のサイズは%uです", (unsigned int)sizeof(piyo));

	return 0;
}
```


ポインタの型の大きさ

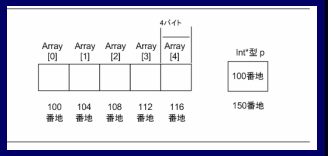
```
#include <stdio.h>

int main(void)
{
	char* hoge;
	int* piyo;

	printf("char*型のサイズは%uです¥n", (unsigned int)sizeof(hoge));
	printf("int*型のサイズは%uです¥n", (unsigned int)sizeof(piyo));

	return 0;
}
```


配列とポインタ入門

```
#include <stdio.h>
int main(void)
{
    int array[5] = {5,10,15,20,25};
    int* p;
    p = &array[0];

    printf("p[0] = %d¥n",*p);
    p++;
    printf("p[1] = %d¥n",*p);
}
```

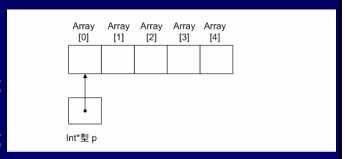

配列とポインタ入門

```
#include <stdio.h>
int main(void)
{
    int array[5] = {5,10,15,20,25};
    int* p;
    p = &array[0];

    printf("p[0] = %d¥n",*p);
    p++;
    printf("p[1] = %d¥n",*p);
}
```



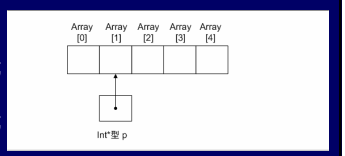
配列とポインタ入門

```
#include <stdio.h>
int main(void)
{
    int array[5] = {5,10,15,20,25};
    int* p;
    p = &array[0];

    printf("p[0] = %d¥n",*p);
    p++;
    printf("p[1] = %d¥n",*p);
}
```



配列とポインタ入門

```
#include <stdio.h>
int main(void)
{
    int array[5] = {5,10,15,20,25};
    int* p;
    p = &array[0];

    printf("p[0] = %d¥n",*p);
    p++;
    printf("p[1] = %d¥n",*p);
}
```



なぜポインタが必要なのか

まとめ

- ポインタは住所（アドレス）を入れる箱である