12

STEP N

포인터

변수
포인터 변수

변수

값을 담기 위한 변수

Bag bag1 = “분홍bag”;

포인터변수

주소를 담기 위한 변수

Bag * bagmemo = &bag1;

114

변수 포인터 변수 선언

Bag bag1 = “분홍bag”;

데이터 타입 | 변수명 | 초기값

Bag *bag2 = &bag1;

데이터 타입 | 변수명 | 주소값

115

Ex1

```c
#include <stdio.h>

int main(){

  int num = 100;
  int * ptr;

  ptr = &num;

  printf("ptr's value\t=> %u \n", (unsigned int)ptr);
  printf("num's addr \t=> %u \n", (unsigned int)&num);

  printf("*ptr's value\t=> %u \n", *ptr);
  printf("num's value \t=> %u \n", num);
}
```

```
ptr's value => 2878057660
num's addr  => 2878057660
*ptr's value    => 100
num's value     => 100
```

116

Ex2-1

두 수를 입력 받고 두 수를 바꾸기(swap) — 포인터 없이

```
num1 & num2 ? 100 200
num1 = 100, num2 = 200
num1 = 200, num2 = 100
```

```c
clude <stdio.h>

int main(){
  int num1, num2, temp;

  printf("num1 & num2 ? ");
  scanf("%d %d", &num1, &num2);

  printf("num1 = %d, num2 = %d\n", num1, num2);
  temp = num1;
  num1 = num2;
  num2 = temp;
  printf("num1 = %d, num2 = %d\n", num1, num2);

  return 0;
}
```

117

Ex2-2

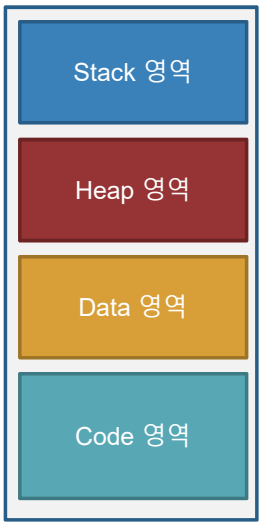
두 수를 입력 받고 두 수를 바꾸기(swap) — 포인터 사용하여

```
num1 & num2 ? 100 200
num1 = 100, num2 = 200
num1 = 200, num2 = 100
```

```c
#include <stdio.h>
void exchange(int * n1, int * n2);
int main(){
  int num1, num2, temp;

  printf("num1 & num2 ? ");
  scanf("%d %d", &num1, &num2);

  printf("num1 = %d, num2 = %d\n", num1, num2);
  exchange(&num1, &num2);
  printf("num1 = %d, num2 = %d\n", num1, num2);
  return 0;
}

void exchange(int * n1, int * n2){
  int temp;
  temp = *n1;
  *n1 = *n2;
  *n2 = temp;
}
```

118

메모리 구조



지역변수, 매개변수, 사용자 정의함수 등

동적할당, 사용자에 의해 할당하고 해제할 수 있는 메모리
할당 : malloc / 해제 : free

전역변수, 정적변수(static으로 선언되는 변수), 프로그램이 종료될 때까지 유지

코드 저장 등

19

정적할당 동적할당

정적할당(static memory allocation)

```
int num1 = 10;
int numarry[10];
char str[20];
```

- 메모리의 크기가 하드코딩 됨
- 프로그램이 실행될 때 메모리 크기가 고정됨
- 프로그램이 종료할 때 메모리가 자동으로 해제됨(장점)
- 프로그램 실행 중 크기 조절이 안됨(단점)

동적할당(dynamic memory allocation)

- 프로그램 실행 중에 **힙 영역**에 메모리 공간을 할당 받음
- 프로그램 실행 중에 함수를 통해 해제할 수 있음
- 실행 중에 할당된 메모리의 크기를 줄이거나 늘일 수 있음
- 더 이상 사용하지 않을 때 메모리를 해제해 주어야 함

```
int *num1 = (int *)malloc(sizeof(int));
char *name = (char *)malloc(sizeof(char)*10);
```

20

메모리 동적할당

프로그램 실행하는 중에 동적으로 메모리를 할당 (힙영역)

정적할당

```
int num = 100;
int *ptr1 = &num;
```

동적할당 `#include <stdlib.h>`

```
int *ptr2 = (int *)malloc(sizeof(int));
*ptr2 = 1234;

. . .

free(ptr2);

 // int numarr[10];
int *ptr3 = (int *)malloc(sizeof(int) * 10);
```

```
#include <stdlib.h>
```

malloc 함수
calloc 함수
realloc 함수
free 함수

구조체 with pointer

```
typedef struct {
  char name[20];
  char address[80];
  int age;
} person;


. . .

person *p1 = (person *)malloc(sizeof(person));

. . .

printf("%s", p1->name);


. . .


free(p1);
```

22

구조체
with pointer

```
이름은? fffg
주소는? asdf
나이는? 10
fffg/ asdf/ 10
```

```c
#include <stdio.h>
#include <stdlib.h>

struct person{
  char name[20];
  char address[80];
  int age;
};
struct person * add_person(){
  struct person * ptr =
      (struct person *)malloc(sizeof(struct person));
  printf("이름은? ");
  scanf("%s", ptr->name);
  printf("주소는? ");
  scanf("%s", ptr->address);
  printf("나이는? ");
  scanf("%d", &(ptr->age));
  return ptr;
}

int main(){
  struct person * p;

  p = add_person();
  printf("%s/ %s/ %d\n", p->name, p->address, p->age);
  free(p);
}
```

메모리 동적할당