동적 메모리

한국기술교육대학교
온라인평생교육원

학습내용

- 동적 메모리 이해
- 동적 메모리 활용

학습목표

- 동적 메모리의 기본 개념을 파악하고 용도를 설명할 수 있다.
- 동적으로 메모리 할당이 필요한 작업에 적용할 수 있다.

동적 메모리 이해

1 개요

특징	정적 메모리	동적 메모리
메모리 할당	컴파일 시간에 이루어짐	실행 시간에 이루어짐
메모리 해제	자동으로 해제	명시적으로 해제
사용 범위	지역 변수는 선언된 블록 내, 전역 변수는 프로그램 전체에서 사용할 수 있음	프로그래머가 원하는 동안만큼 사용할 수 있음
메모리 관리	컴파일러의 책임	프로그래머의 책임

1 데이터의 개수를 미리 알 수 없을 때 사용

2 처리 대상 데이터가 유동적일 때, 특히 변동 폭이 큰 경우

```
int size;

scanf("%d", &size);

int arr[size];
```

배열의 크기를 입력받음

컴파일 에러

동적 메모리 이해

2 라이브러리 함수

1 종류

1 헤더파일 stdlib.h

함수	void* malloc (size_t size);
설명	• 실행 시 메모리를 할당(초기화 없음) • size : 할당 크기
반환	• 할당된 메모리 포인터를 반환 • 할당 실패 시 null을 반환

함수	void* calloc (size_t num, size_t size);
설명	• 실행 시 메모리를 할당(초기화 0) • num : 개수 • size : 기본 크기
반환	• 할당된 메모리 포인터를 반환 • 할당 실패 시 null을 반환

동적 메모리 이해

2 라이브러리 함수

1 종류

1 헤더파일 stdlib.h

함수	void* realloc (void* ptr, size_t size);
설명	• 할당된 메모리의 크기를 변경 • ptr : 재할당 메모리 포인터 • size : 재할당 크기
반환	• 재할당된 메모리 포인터를 반환 • 할당 실패 시 null을 반환

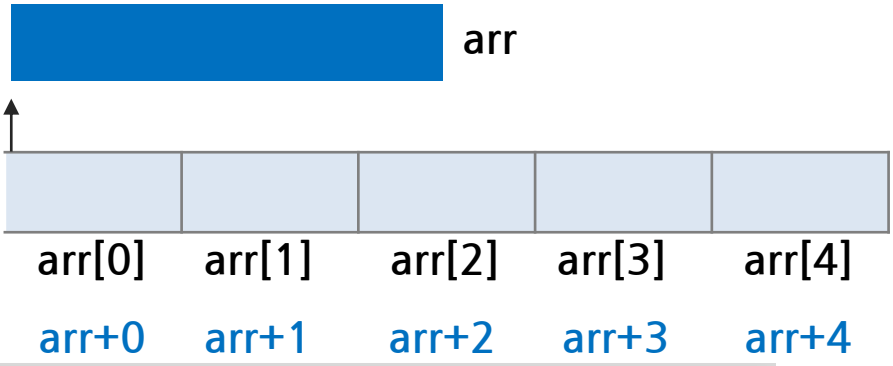
함수	void free (void* ptr);
설명	• 할당된 메모리를 해제 • ptr : 해제할 메모리 포인터
반환	• none

동적 메모리 이해

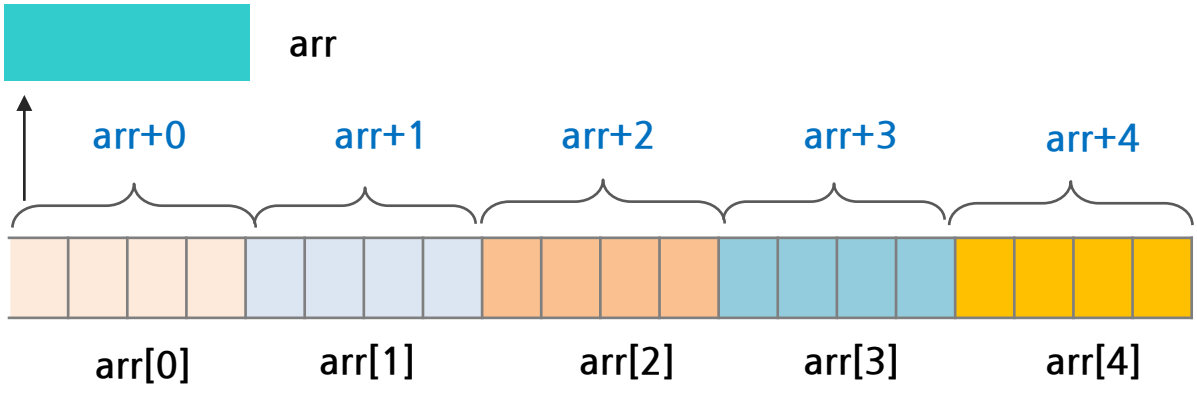
2 라이브러리 함수

2 malloc()

```
char* arr;
arr = malloc( 5 );
```




```
int* arr;
arr = (int *) malloc( 20 );
```



동적 메모리 이해

2 라이브러리 함수

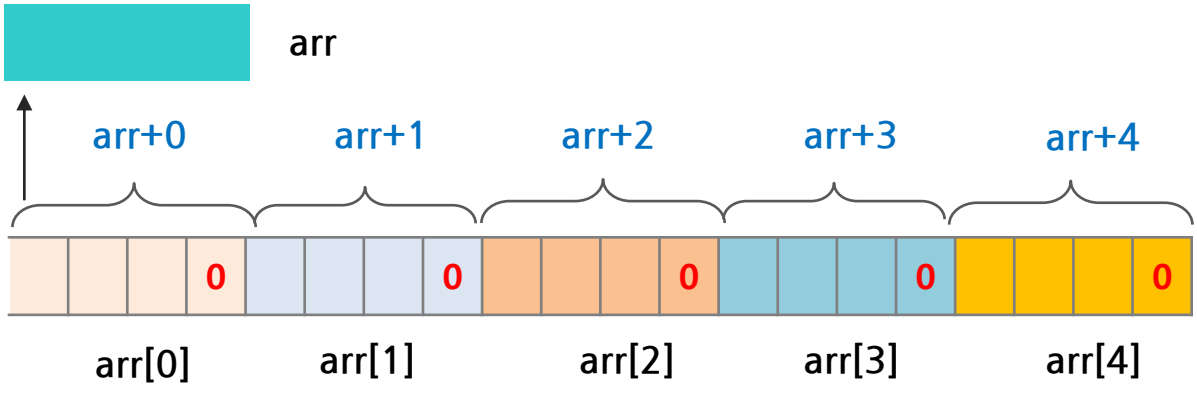
3 free()

```
int* arr;
arr = (int *) malloc( 5 );


free(arr);
```

4 calloc()

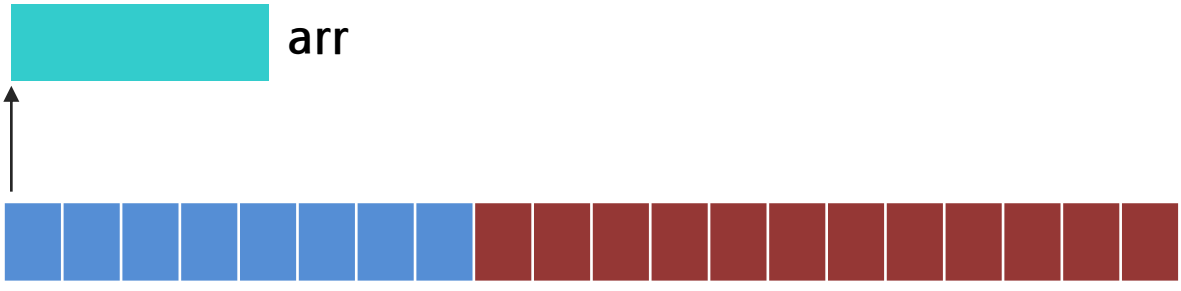
```
int* arr;
arr = (int *) calloc( 5 , 4 );
```



동적 메모리 이해

5 realloc()

```
int* arr;
arr = (int *) malloc( 8 );
arr = (int *)realloc(arr, 20)
```



동적 메모리 활용

1 함수 기초

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

int main()
{
        char *a;
        int size;
        scanf("%d", &size);
        a = malloc( sizeof(char)*size );
        strcpy(a,"hi");
        printf("문자수 : %d   문자열 : ₩n", strlen(a), a );
        free(a);
        return 0;
}
```

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

int main()
{
        int *a;
        int size;
        scanf("%d", &size);
        a = (int *) calloc( sizeof(int), size );

        free(a)
        return 0;
}
```

동적 메모리 활용

2 함수 활용

Q 문자열을 입력받고 하나의 동적 메모리에 계속 붙여서 저장하는 프로그램을 작성하시오. (“end” 입력 시 입력 종료)

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

int main()
{
    char *a, str[20];
    a = (char *)calloc(1,1);
    if (a == NULL) printf("Fail Allocation");
    while(1){
                    gets(str);
                    if( !strcmp(str,"end") )
                        break;
                    a = (char *)realloc(a,strlen(str)+1);
                    strcat(a,str);
    }
    printf("₩n%s", a);

    free(a);
    return 0;
}
```

학습정리

1. 동적 메모리 이해

- 동적 할당은 실행 시에 할당되는 메모리임
- 동적 할당은 힙 영역에 할당함
- 동적 할당은 실행 시 크기가 정해지는 데이터 처리에 효과적임
- 동적 할당된 공간은 프로그래머가 해제해야 함

2. 동적 메모리 활용

- malloc, calloc은 동적으로 메모리를 할당하는 라이브러리 함수임
- calloc은 동적 할당 후 0으로 초기화
- free는 동적 할당된 메모리를 해제함
- realloc은 동적 메모리의 크기를 변경하여 할당하는 것이 가능함