# 〈C 프로그래밍 기말고사 및 코테 대비〉

## 개념편

중간 고사때는 잡 설명이 많았다. 이번에는 나올만한 개념과 반드시 알아야하는 개념만 설명하고 넘어갈거다. 교수님이 주신 자료 기반으로 부가 설명하는 식으로 진행했으니 보다 이해하기 편할거다.. 파이팅~!

### [배열]

#### 1. 배열

배열은 여러 개의 변수를 나란히 연결하는 개념이다. (제발 그냥 받아드리세요...ㅜ 그래도 설명해볼게 한 1000개의 변수가 필요하면 변수 이름 만드는게 더 어려울거잖아. 그니깐 배열이란걸 만들었다고 이해하고 중요한건 이걸 이용한 문제풀이니깐 그냥 받아드려). 2차원 배열까지는 뭐 괜찮다고 쳐도 3차원 배열같은거 나오면 어지러울거다. 수업 시간에 2차원 배열까지 밖에 안해서 아마도 안 나올거 같지만 3차원 배열 문제같은거 궁금하다면 구글링해서 3차원 배열 검색하면 예제 많이 나오니깐 참고하면 좋을 거 같다. 배열을 사용하는 방법은 데이터형식 배열이름[개수]; 이런식으로 쓰면 된다.

### 1. 배열을 사용하는 이유

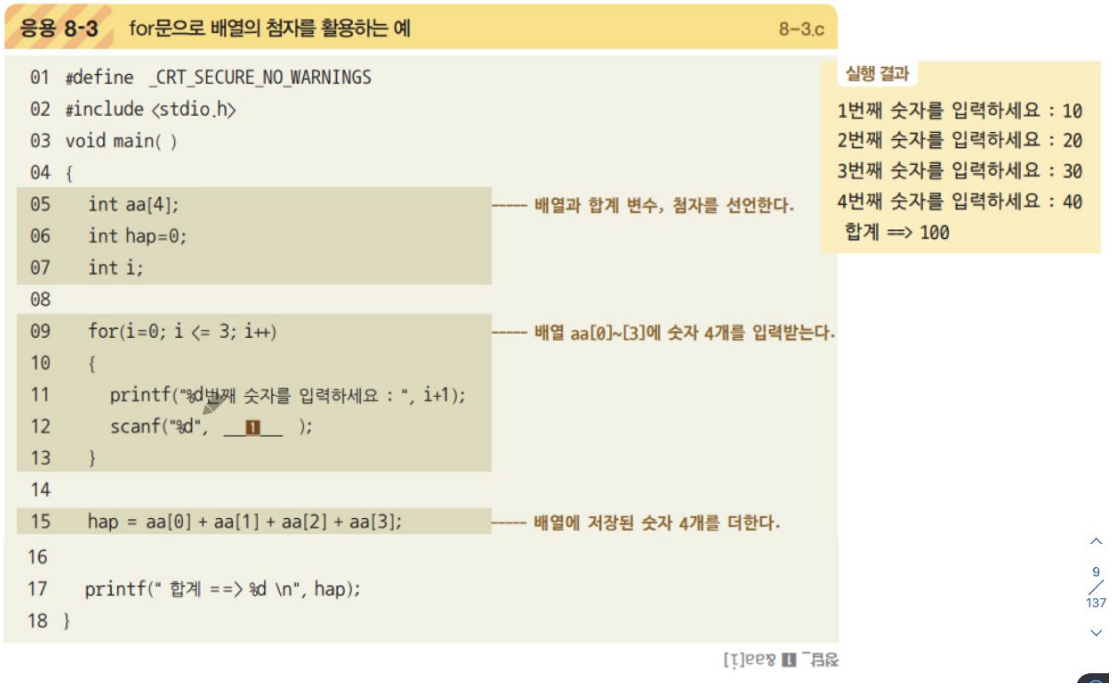
**기본 8-2** 배열에 값을 선언하여 출력하는 예 — 8-2.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int aa[4];                    ---- 정수형 배열을 선언한다.
06    int hap;
07
08    printf("1번째 숫자를 입력하세요 : ");
09    scanf("%d", &aa[0]);          ---- aa[0]에 숫자를 입력한다.
10    printf("2번째 숫자를 입력하세요 : ");
11    scanf("%d", &aa[1]);          ---- aa[1]에 숫자를 입력한다.
12    printf("3번째 숫자를 입력하세요 : ");
13    scanf("%d", &aa[2]);          ---- aa[2]에 숫자를 입력한다.
14    printf("4번째 숫자를 입력하세요 : ");
15    scanf("%d", &aa[3]);          ---- aa[3]에 숫자를 입력한다.
16
17    hap = aa[0] + aa[1] + aa[2] + aa[3];   ---- 입력받은 배열에 저장된 숫자의 합계 결과이다.
18
19    printf(" 합계 ==> %d \n", hap);
20 }
```

실행 결과

```
1번째 숫자를 입력하세요 : 10
2번째 숫자를 입력하세요 : 20
3번째 숫자를 입력하세요 : 30
4번째 숫자를 입력하세요 : 40
 합계 ==> 100
```

배열이 왜 편한가를 생각하면 배열변수명[i]라고 한 후 for문을 돌려버리면 값을 저장하고 출력하는 것이 굉장히 편하다고 할 수 있다. 이게 생각보다 활용도가 높으니 자주 연습해서 익숙해지는걸 추천한다.



응용 8-3 for문으로 배열의 첨자를 활용하는 예 8-3.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <stdio.h>
03 void main( )
04 {
05    int aa[4];                          ---- 배열과 합계 변수, 첨자를 선언한다.
06    int hap=0;
07    int i;
08
09    for(i=0; i <= 3; i++)               ---- 배열 aa[0]~[3]에 숫자 4개를 입력받는다.
10    {
11       printf("%d번째 숫자를 입력하세요 : ", i+1);
12       scanf("%d", &aa[i] );
13    }
14
15    hap = aa[0] + aa[1] + aa[2] + aa[3];  ---- 배열에 저장된 숫자 4개를 더한다.
16
17    printf(" 합계 ==> %d \n", hap);
18 }
```

실행 결과

```
1번째 숫자를 입력하세요 : 10
2번째 숫자를 입력하세요 : 20
3번째 숫자를 입력하세요 : 30
4번째 숫자를 입력하세요 : 40
 합계 ==> 100
```



배열에 값을 저장하는 방법은 그냥 {}(블록)안에 ,(콤마)를 사용해서 넣어주면 되는데 배열의 0번째 순서부터 차례대로 넣어주면 된다. 여기서 배열 개수보다 조금 넣어주는건 괜찮은데 더 많이 넣으면 오류 뜨는걸 알아둬야한다.(시험 문제 나올만함) 배열에 모든 것을 0으로 초기화 하고 싶을때는 int aa[1000] = {0}; 이렇게 해주면 된다.(다른 숫자로 모두 초기화는 불가능 -> int aa[1000] = {15}; 이런 식으로 하면 15하고 뒤에는 모두 0으로 출력된다.) 그리고 짝수로 초기화하는 것과 역순으로 초기화 하는 방법을 알려준 것으로 보아(간단히 언급했지만) 이런것을 이용한 심화문제가 출제 될 수도 있으니 이해해두자.

```
// 배열 aa를 짝수로 초기화
for(i = 0; i < 100; i++)
    aa[i] = i * 2;

//배열 bb를 배열 aa의 역순으로 초기화
for(i = 0; i < 100; i++)
    bb[i] = a[99-i];
```

그리고 배열의 크기 sizeof()함수를 사용해서 배열의 개수 구하는 문제는 자주 나오니 꼭 공부해두자.

▪ **배열의 크기 알아내기**

**기본 8-6** 배열의 크기를 계산하는 예 8-6.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int aa[] = {10, 20, 30, 40, 50};   ---- 배열을 선언한다. 변수의 개수를 지정하지 않았다.
06    int count;                         ---- 배열 크기를 저장할 변수이다.
07
08    count = sizeof(aa) / sizeof(int);  ---- 배열 크기를 계산한다.
09
10    printf("배열 aa[]의 요소의 개수는 %d 입니다.\n", count);
11 }
```

실행 결과

```
배열 aa[]의 요소의 개수는 5 입니다.
```

->저기서 sizeof(aa)는 4바이트짜리 5개니깐 20이고 sizeof(int)는 4바이트 1개로 4를 의미하니깐 20/4해서 aa의 배열 개수가 5개라는 것을 알아낼 수 있다.

-여기까지가 9-1

**2. 배열과 문자열**

문자열 배열에서는 NULL문자(₩0)을 반드시 생각해줘야한다. 널문자를 포함 시켜야하기 때문에 문자열 배열 크기를 설정할때 원하는 문자열의 개수보다 1개 크게 설정해야한다.(이거 문제에서 나올만함) 그리고 여기서 나오는게 문자와 문자열의 차이이다. 문자형 변수(c)는 저장할때 ‘ ’이걸로 저장하고 문자열형 변수(s)는 “ ”이걸로 저장한다는 것을 알고 있어야한다. 그리고 문자를 저장할때는 반드시 한 글자만 올 수 있다.

▪ **문자열의 기본 형식**

**여기서 잠깐 올바른 문자 표현**

- 문자는 반드시 ‘ ’로 감싸야 하며 한 글자만 올 수 있으며 ❶~❸은 모두 틀린 표현
- char a;
- ❶ a = ‘Ab‘; ❷ a = ”A”; ❸ a = “Ab”;

strcpy() 함수가 나오는데 이는 문자열 저장하는 도구이다. 다른 데이터 형식은 대입 연산자를 사용해서 대입했으나 문자열은 strcpy() 함수를 사용할 수 있다. 이때 string.h 라이브러리 선언을 해줘야한다. 뒤에서 자세하게 다룬다.

### ▪ 문자열의 기본 형식

**기본 3-14** 문자열 형식 사용 예 1 — 3-14.c

```
01 #define _CRT_SECURE_NO_WARNINGS
02 #include <string.h>
03 #include <stdio.h>
04 void main( )
05 {
06    char str1[10];                  ---- 문자형 배열 str1과 str2를 선언한다.
07    char str2[10];
08    char str3[10] = "CookBook";     ---- 문자형 배열 str3을 선언함과 동시에 문자열을 대입한다.
09
10    strcpy(str1, "Basic-C");        ---- str1에 문자열을 대입한다.
11    strcpy(str2, str3);             ---- str3의 값을 str2에 복사한다.
12
13    printf("str1 ==> %s \n", str1); ---- 문자형 배열 str1, str2, str3을 출력한다.
14    printf("str2 ==> %s \n", str2);
15    printf("str3 ==> %s \n", str3);
16 }
```

실행 결과

```
str1 ==> Basic-C
str2 ==> CookBook
str3 ==> CookBook
```

문자열과 배열의 뒷이야기라는 내용으로 교수님이 설명하신 내용이 있는데 어려운건 아니고 배열의 개수와 배열에 들어가는 수가 같을때 어떻게 되는가? 그리고 10개 배열만 만들었는데 막 50번째 배열을 출력하면 어떻게 되는지에 관한 내용인데 그냥 당연히 쓰레기값이 들어갔으니깐 아무거나 출력되게된다. 여기서 강조하신건 널문자가 없으면 쓰레기 값으로 채워지며 끝까지 출력된다는 것이다. 코드보면 이해 될 것이다.

main.c

```c
#include <stdio.h>

int main(void) {
  //10개 배열에 10개 다 채워서 \0(널문자)가 못 들어간 상황
  char str[10] = "0123456789";

  printf("str ==> %s\n",str);//널문자 만날때까지 출력하기 때문에 9까지 나오고 오류
  printf("str[7] ==> %c\n",str[7]);//str[7]에는 7들어가있으니깐 7출력
  printf("str[50] ==> %c\n",str[50]); //str[50]에는 쓰레기값있으니깐 아무거나 나옴

  return 0;
}
```

Console / Shell

```
> make -s
> ./main
str ==> 0123456789kh
7
@
>
```

**<strlen() 함수> -> 문자열 배열의 개수를 구하는 도구 //string+length**

문자열을 입력받고 반대로 출력하는 프로그램을 짠다고 했을때 가장 중요한 것이 입력받은 문자열의 길이가 몇인지에 대한 내용이다. 그래야지 for문을 몇번 돌려야할지 알 수 있으니깐. strlen() 함수는 string.h라이브러리에 포함되어있고 이 함수는 널문자를 제외한 갯수를 출력해주니깐 이것도 생각해야한다. 예제 같이 사용하면 된다.

- **문자열의 길이를 알려주는 함수: strlen( )**

**기본 8-9** 문자열 처리 함수 strlen( ) 사용 예 — 8-9.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <string.h>          ---- 문자열 함수의 목록이 있는 string.h를 포함한다.
03 #include <stdio.h>
04 void main( )
05 {
06    char ss[] = "XYZ";        ---- 문자열 배열과 길이를 저장할 변수이다.
07    int len;
08
09    len = strlen(ss);         ---- 문자열 배열 ss의 길이를 구한다.
10
11    printf("문자열 \"%s\"의 길이 ==> %d \n", ss, len);
                                      ---- 큰따옴표의 내용을 출력하기 위해 \" 문자를 사용한다.
12 }
```

실행 결과

```
문자열 "XYZ"의 길이 ==> 3
```

**<strcpy() 함수> -> 배열을 복사할 때 사용하는 도구 //string + copy**

배열을 복사할 때 사용하는 함수이다. strcpy(문자열 배열 A , 문자열 B); 이런 식으로 사용하면 된다. 이 함수는 이미 선언된 문자열 배열에 다른 문자열을 대입하고 싶을 때 주로 사용한다. 주의해야할 것은 배열 개수이다. 복사하고 싶은 문자열 배열의 크기가 기존에 있던 문자열보다 작으면 오류가 뜨니 주의 해야한다.

**기본 8-10** 문자열 처리 함수 strcpy( ) 사용 예 — 8-10.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <string.h>
03 #include <stdio.h>
04 void main( )
05 {
06    char ss[4];               ---- 문자열 배열을 선언한다.
07
08    strcpy(ss, "XYZ");        ---- 배열 ss에 문자열 "XYZ"를 복사한다.
09
10    printf("문자열 ss의 내용 ==> %s \n", ss);
11 }
```

실행 결과

```
문자열 ss의 내용 ==> XYZ
```

**<strcat() 함수> -> 두 문자를 이어줄 때 사용하는 도구**

두 문자열을 이어줄때 사용하는 함수이다. 앞서 작성한 배열이 있는데 뒤에 뭔가 이어서 넣어주고 싶을 때 사용한다. 이어주는 자리는 널 문자 자리부터 시작한다.(널문자가 자동으로 뒤로 감)

**기본 8-11** 문자열 처리 함수 strcat( ) 사용 예 8-11.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <string.h>
03 #include <stdio.h>
04 void main( )
05 {
06    char ss[7] = "XYZ";                  ---- 문자열 배열을 선언하고 초기화한다.
07
08    strcat(ss, "ABC");                   ---- 배열 ss의 내용("XYZ")에 문자열 "ABC"를 이어서 다시 ss에 대입한다.
09
10    printf("이어진 문자열 ss의 내용 ==> %s \n", ss);
11 }
```

실행 결과

```
이어진 문자열 ss의 내용 ==> XYZABC
```

**<strcmp() 함수> -> 두 문자열을 비교하는 도구**

두 문자열을 비교하는 함수이다. strcmp(문자열 A, 문자열 B); 라고 하면 A-B의 결과를 돌려준다. 결과가 0이 나오면 같은 문자열이라는 뜻이다. 이 함수는 두 문자열이 같은지 판단할 때만 주로 쓴다.(문제에서 자주 씀.) -> 11-1의 예제 04문제에서 확인해보자.

**<gets() 함수와 puts() 함수> -> get+string, put+string**

문자열을 입력받고 출력할때 사용한다. gets()은 scanf()와 같은 용도이고 puts()은 printf()와 같은 용도이다.

- **문자열 입출력 함수**

**응용 8-13** 문자열 입출력 함수 gets( ), puts( ) 사용 예 8-13.c

```
01 #define  _CRT_SECURE_NO_WARNINGS
02 #include <string.h>
03 #include <stdio.h>
04 void main( )
05 {
06    char ss[20];                         ---- 문자형 배열 ss와 tt를 선언한다.
07    char tt[20];
08    int r1, r2;
09
10    puts("첫 번째 문자열을 입력하세요.");   ---- 배열 ss와 tt에 문자열을 입력한다.
11    gets_s(ss, sizeof(ss));
12
13    puts("두 번째 문자열을 입력하세요.");
14    gets_s(tt, sizeof(tt));
15
16    r1 = strlen(ss);                     ---- 배열 ss와 tt의 문자열 길이를 저장한다.
17    r2 = strlen(tt);
18
```

```
19    printf("첫 번째 문자열의 길이 ==> %d \n", r1);
20    printf("두 번째 문자열의 길이 ==> %d \n", r2);
21
22    if (strcmp(ss, tt) == 0)
23      puts("두 문자열의 내용이 같습니다.\n");
24    else
25      puts("두 문자열의 내용이 다릅니다.\n");
26  }
```

각 배열의 문자열 길이를 출력한다.

ss와 tt의 문자열이 같은지 비교한다.

```
실행 결과
첫 번째 문자열을 입력하세요.
IT CookBook
두 번째 문자열을 입력하세요.
Hanbit
첫 번째 문자열의 길이 ==> 11
두 번째 문자열의 길이 ==> 6
두 문자열의 내용이 다릅니다.
```

-여기까지가 10-1 위에 배운 함수들을 쓰려면 반드시 string.h 라이브러리 선언해야한다. 혹시나 해서 말하는데 string은 문자열이라는 의미이다.

11-1 배열 실습은 코딩연습편에서 다루도록 하겠다.

### 3. 다차원 배열

다차원 배열에는 뭐 2차원 3차원 보이지는 않지만 가상으로 4차원도 만들 수는 있다. 우리 수업때는 2차원만 다뤘고 3차원은 아마도 안나올거 같지만 나올 수도 있으니 이해는 반드시 해두는 것을 추천한다. 선언할때는 데이터형식 배열변수명[가로 개수][세로 개수]; 이런식으로 선언하면 된다.

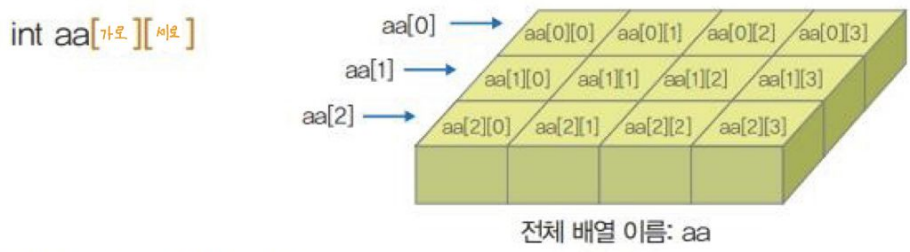



그림 8-16 2차원 배열의 개념

값을 입력 받을때는 중첩 for문을 사용해서 넣어주면 된다.

**응용 8-15** 2차원 배열 사용 예 2 — 8-15.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int aa[3][4];                    ---- 2차원 배열과 첨자 변수를 선언한다.
06    int i, k;
07
08    int val=1;                       ---- 배열에 들어갈 값을 초기화한다.
09
10    for( i=0; i < 3; i++ )           ---- 바깥 for문을 세 번 반복한다.
11    {                                     즉 앞 첨자가 행 단위로 변경된다.
12       for (k=0; k < 4; k++)         ---- 안쪽 for문을 네 번 반복한다.
13       {                                  즉 뒤 첨자가 열 단위로 변경된다.
14          aa[i][k] = val;            ---- 배열에 val 값을 입력한 후 1 증가시킨다.
15          val++;
16       }
17    }
18
19    printf("aa[0][0]부터 aa[2][3]까지 출력 \n");
20
21    for( i=0; i < 3; i++ )           ---- 입력과 동일한 개념으로 12회 출력한다.
22    {
23       for( k=0; k < 4; k++ )
24       {
25          printf("%3d ", aa[i][k] );
26       }
27       printf("\n");                 ---- 한 행을 출력한 후 줄을 넘긴다.
28    }
29 }
```

정답_ 1 k=0; k<4; k++ 2 aa[i][k]

실행 결과

```
aa[0][0]부터 aa[2][3]까지 출력
 1  2  3  4
 5  6  7  8
 9 10 11 12
```

3차원 배열은 2차원 배열의 초기화를 한 번 더 하는 개념이다. 콤마로 분리하고 전체를 다시 블록으로 묶으면 된다.

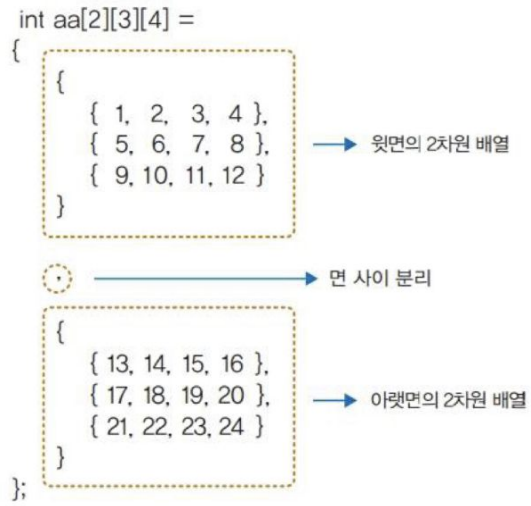



그림 8-19 3차원 배열의 초기화

-여기까지가 11-2의 내용이고 예제들은 코딩 연습편에서 다루도록 하겠다.

4. 스택

스택은 자료구조중에 하나이다. 데이터쪽에서 주로 다루고 깊게 공부하고 싶으면 데이터베이스를 공부하는 것을 추천한다. 개념은 가장 먼저 들어간 것이 가장 나중에 나오는 것을 의미한다. 세워져 있는 물병을 눕힌다고 생각하면 이해하기 편할 것이다. 스택에는 top, push, pop이 있다. top은 가장 마지막에 들어간 데이터의 위치를 가리킨다. push는 데이터를 넣는 것이고 pop은 데이터를 빼는 것이다. 어렵게 생각할 수 있는데 그냥 이 개념을 생각하고 그대로 만들어버리면된다.

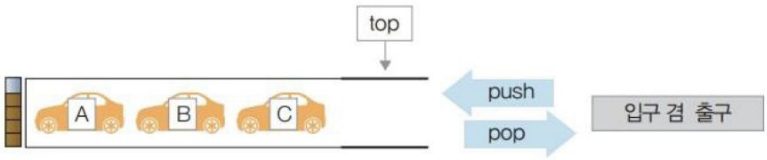
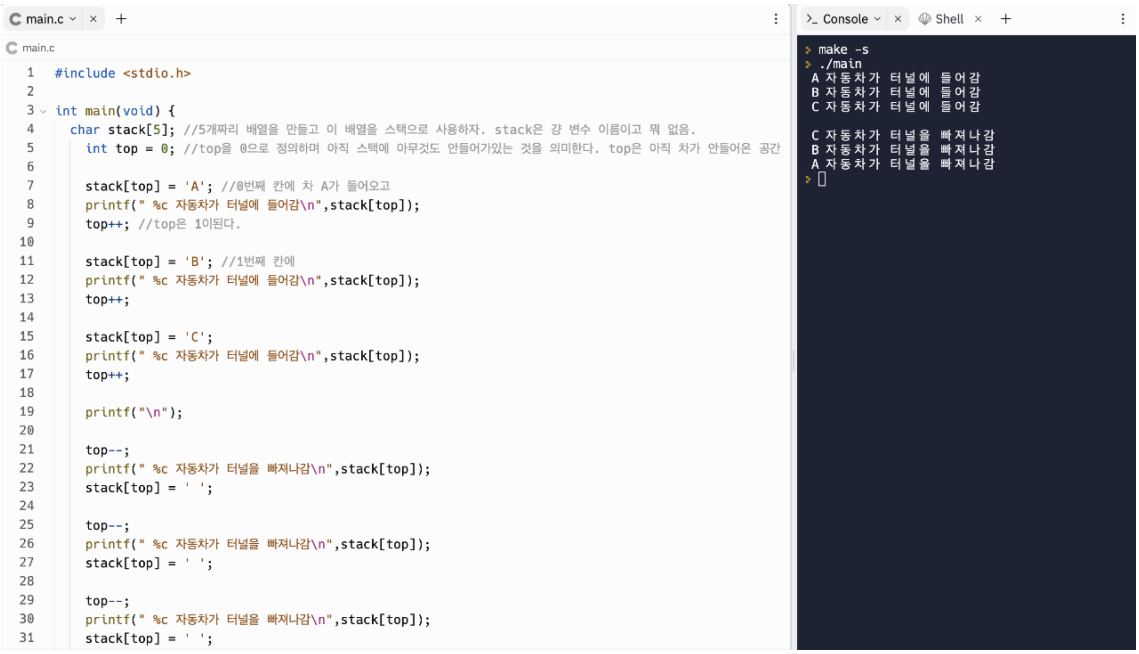



그림 9-1 스택의 기본 개념

예제를 보면




-여기까지가 12-1

## [메모리 주소와 포인터]

### 1. 정수형 변수의 메모리 할당

메모리는 바이트(byte) 단위로 나뉘고 각 바이트에는 주소가 지정되어있다. 예를 들어보면 정수형 변수 크기는 4바이트니깐 int a; 선언을 하면 임의의 위치에 4바이트가 자리잡혀있다. 주소를 알려면 &을 붙이면 된다.

**기본 9-3** 변수의 주소 알아내기 9-3.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int a = 100;
06    int b = 200;
07
08    printf("변수 a의 주소는 %d 입니다. \n", &a);
09    printf("변수 b의 주소는 %d 입니다. \n", &b);
10 }
```

---- a와 b의 주소를 출력한다.

실행 결과

```
변수 a의 주소는 20184760 입니다.
변수 b의 주소는 20184748 입니다.
```

## 2. 정수형 배열의 메모리 할당

정수형 배열에서 한 칸은 4바이트를 의미한다. 따라서 주소를 표현할때 배열의 첨자로 표현하거나 배열의 이름으로 표현할 수 있다.

- aa+1에서 +1은 단순히 1을 더하라는 의미가 아니라

  **'배열 aa의 위치에 서 한 칸 건너뛰라'는 의미**

- '한 칸'은 현재 aa가 정수형 배열이므로 4바이트를 의미

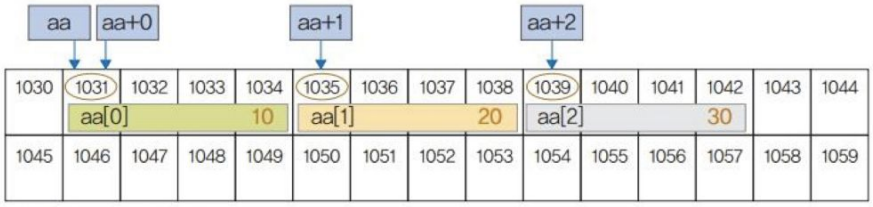



그림 9-8 배열 이름으로 주소 표현

표 9-1 배열의 주소 표현

| 배열 첨자로 표현 | 배열 이름으로 표현 | 실제 주소 |
|---|---|---|
| &aa[0] | aa + 0 | 1031 |
| &aa[1] | aa + 1 | 1035 |
| &aa[2] | aa + 2 | 1039 |

예제로 보면

**응용 9-5** 정수형 배열의 메모리 할당 2 — 9-5.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    int aa[3] = {10, 20, 30};
06
07    printf("&aa[0]는 %d, aa+0은 %d \n", &aa[0], aa+0);   ----- &aa[0]==aa+0
08    printf("&aa[1]는 %d, aa+1은 %d \n", &aa[1], aa+1);   ----- &aa[1]==aa+1
09    printf("&aa[2]는 %d, aa+2는 %d \n", &aa[2], aa+2);   ----- &aa[2]==aa+2
10 }
```

실행 결과

```
&aa[0]는 7601340, aa+0은 7601340
&aa[1]는 7601344, aa+1은 7601344
&aa[2]는 7601348, aa+2는 7601348
```

### 3. 포인터

포인터는 어떤 변수의 주소를 저장하는 변수이다. 포인터라는게 pointer 즉 가르키는 것을 의미한다. 포인터 변수를 선언하고 포인터 변수에 자료형 변수의 주소를 저장하면 포인터 변수가 자료형 변수가 있는 주소를 가르키고 있는 것이다. 포인터 변수를 선언할때는 아스타리카(*)를 사용해서 선언한다. 선언할때 -> 자료형 *변수명; or 자료형* 변수명; 이런식으로 선언하면 된다. 그리고 포인터 변수에 자료형 변수의 주소를 저장할때는 주소연산인 앰퍼센트(&)를 사용해서 저장하면 된다.

**기본 9-6** 일반 변수와 포인터 변수의 관계 — 9-6.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    char ch;                  ---- 문자형 변수와 포인터 변수를 선언한다.
06    char* p;
07
08    ch = 'A';                 ---- 문자 'A'를 ch에 대입하고 ch의 주소를 p에 대입한다.
09    p = &ch;
10
11    printf("ch가 가지고 있는 값: ch  ==> %c \n", ch);
12    printf("ch의 주소(address): &ch ==> %d \n", &ch);
13    printf("p가 가지고 있는 값 : p   ==> %d \n", p);
14    printf("p가 가리키는 곳의 실제 값 : *p  ==> %c \n", *p);
15 }
```

실행 결과

```
ch가 가지고 있는 값: ch  ==> A
ch의 주소(address): &ch ==> 9042587
p가 가지고 있는 값 : p   ==> 9042587
p가 가리키는 곳의 실제 값 : *p  ==> A
```

포인터 변수에 저장되어있는 주소의 변수를 포인터 변수로 가져올 수 있는데 이를 역참조라고 한다.(수업시간에 배웠는데 역참조라고는 안배웠음. 근데 역참조라고 부름) 역참조는 '*포인터_변수' 이런식으로 사용한다.

main.c

```c
#include <stdio.h>

int main(void) {
  int a=10; //a를 10으로 초기화하고
  int *p; //포인터 변수 p를 정수형으로 선언

  p=&a; //포인터변수 p에 a의 주소를 저장

  printf("a : %d\n",a); //변수 a값을 출력
  printf("*p : %d\n",*p); //포인터 변수 p를 역참조하여 a에 변수를 출력
  printf("*(&a) : %d\n",*(&a)); //&a는 p를 의미하기에 위와 같은 의미
  //즉 a==*p==*(&a)이다.
  return 0;
}
```

Console / Shell

```
make -s
./main
a : 10
*p : 10
*(&a) : 10
```

-여기까지가 12-2

## 4. 배열과 포인터의 관계

(1)문자형 배열과 포인터

일단 변수들의 주소를 저장할때는 주소연산자인 앰퍼스트(&)를 사용해서 저장했었다. 하지만 문자열로된 배열에서는 다르다. 만약 char a[4] = “xyz”; char *p; 이런식으로 선언했을때 포인터 변수 p에 문자열 배열을 다 넣을 수는 없다. 따라서 첫번째 주소를 저장하고 포인터 변수+n(임의에 숫자)으로 a[0]이상의 배열의 주소를 저장한다. 어려울 수 있는데 조금만 생각해보면 문자열 s는 1바이트가 아니지만 문자 c는 1바이트이고 주소는 1바이트 단위로 세기 때문에 포인터 변수+n(임의에 숫자)으로 사용한다. 배열에서 중요한 것은 선언이다. 주소연산자를 사용하지 않고 포인터 변수에 저장해야한다. 왜그럴까? 배열의 이름은 그 자체로 주소이기 때문이다.

**기본 9-8** 문자형 배열과 포인터의 관계 1 — 9-8.c

```
01 #include <stdio.h>
02
03 void main( )
04 {
05    char s[8]= "Basic-C";          ------ 문자형 배열을 선언하고 초깃값을 대입한다.
06    char  *p;                      ------ 문자형 포인터 변수를 선언한다.
07
08    p = s;                         ------ p에 배열 s의 주소를 대입한다.
09
10    printf("&s[3] ==> %s\n", &s[3]);   ------ 문자열과 포인터의 주솟값을 %s로 출력한다.
11    printf("p+3   ==> %s\n\n", p+3);
12
13    printf("s[3]   ==> %c\n", s[3]);   ------ 문자와 포인터의 실제 값을 %c로 출력한다.
14    printf("*(p+3) ==> %c\n", *(p+3));
15 }
```

실행 결과

```
&s[3] ==> ic-C
p+3   ==> ic-C

s[3]   ==> i
*(p+3) ==> i
```

→ 이 예제에 중요한 개념이 들어있다. printf("%s", &s[3]);의 의미는 구문은 해당 포인터부터 시작하여 널 종료 문자(₩0)가 나올 때까지 문자열을 출력하라는 의미라는 것이다.

-여기까지 13-1

교수님과 조교님이 말씀하시길 함수, 포인터, 배열 이 세가지를 섞어서 문제를 만드시겠다고 하셨다. 근데 생각해보면 저거 3개 섞으면 부분점수 주기도 편하고 어렵게 안내도 알아서 어려워지니깐 당연히 섞을거라고 생각하고 연습하는걸 추천한다. 포인터가 안들어가 있는 예제에 포인터 개념을 써서 역참조하면서 푸는 것을 연습하고 모든 예제의 알고리즘을 함수화 시켜보는 것 또한 추천한다. 글고 위에서 3차원 배열 나올 수도 있다고 했는데 안나온다는거 같고 배열 포인터... 제발 깊은 이해와 연습 부탁합니다.