자료구조와 프로그래밍

◆ 자료 타입(Data Type)

- 기본 자료 타입

  자료 형태 + 연산

  int, foat, double, char. . .

  * java 등 고급 프로그램 언어에서 제공하는 자료타입을 알아보자.

- 군집 자료 타입

  배열(array)

  기본자료타입의 데이터가 여러 개인 경우

- 사용자 정의 자료 타입(구조체)

  C언어에서 제공하는 사용자 정의 자료 타입의 선언과 활용에 대하여 알아보자.

배열의 개념

배열 : 자료타입이 모두 같고 같은 이름으로 **참조**되는 데이터들의 집합

- 각 데이터들은 메모리 안에 인접한 위치를 차지한다.
- 배열의 각 요소를 구별하기 위하여 첨자(subscript) 사용

  cf. 구조체 : 서로 다른 데이터 형의 데이터를 묶어놓은 집합(레코드개념)

배열의 선언

type var_name[size]

예) int a[20];

이때 프로그램에서 색인(첨자)을 이용하여 20개의 각 변수를 사용

a[0], a[1], ... , a[19]

배열의 개념

배열의 대표적인 정보

배열의 이름 = 배열의 첫데이터의주소

a = &a[0]

a+i = &a[i]

***(a+i) = a[i]**

배열변수의 사용

첨자를 이용하여 각 변수를 처리하기 위하여 for문으로 데이터처리

```
sum = 0;
for (i=0; i < n; i++)
        sum = sum + a[i];
```

배열의 첨자 : for 반복문의 lcv

배열의 주소 계산

◆ 배열(array)

- 배열의 주소 계산 : 배열의 데이터가 연속적으로 저장되어 있으므로 자동 계산 가능

  int a[5] = {10, 20, 30, 40, 50}

  a =&a[0] = 1000 번지, sizeof(int) =2

  a[3] = 1006 번지

변수	메모리 주소
a[0]	a
a[1]	a + 1* sizeof(int)
a[2]	a + 2 * sizeof(int)
a[3]	a + 3 * sizeof(int)
a[4]	a + 4 * sizeof(int)

배열의 주소 계산

◆ 배열(array)

• 행 우선 저장방식 – C language

int b[4][3]

int b[d_1] [d_2]

	0	1	2
0	101	107	109
1	221	231	251
2	311	341	351
3	400	405	477

b[0][0]	101
b[0][1]	107
b[0][2]	109
b[1][0]	221
b[1][1]	231
b[1][2]	251
b[2][0]	311
b[2][1]	341
b[2][2]	351
b[3][0]	400
b[3][1]	405
b[3][2]	477

배열의 주소 계산

◆ 배열(array)

- 2차원 배열의 주소

  int b[d_1] [d_2]

  -행 우선방식 C경우

  b[i][j]의주소 = b + {i * d_2 + j} * s

  s : 데이터 1개가 차지하는 메모리 소자 수

	0	1	2
0	101	107	109
1	221	231	251
2	311	341	351
3	400	405	477

- 예) C의 경우 int b[4][3]에서

  &b[0][0] = 80번지에 저장. b[3][1] 의 주소는?

  (단, 정수데이터는 메모리 소자2개 차지 가정)

  80 + {(3 * 3 + 1} * 2 = 100

확인문제

C 프로그램에서 float a[10][20] 로 선언된 배열 a의 첫 원소 a[0][0]는 200번지에 저장되어 있으며 float 하나는 메모리 소자 네 개를 차지한다고 할 때 a[6][12]의 주소는 몇 번지인가? 또 몇 번째 데이터인가?

200 + {(6-0) * 20 + (12-0)} * 4 = 728

1 + (6*20) + 12 = 133 째 데이터

배열의 주소 계산

◆ 배열(array)

- **열 우선 저장방식 – Fortran**

integer M(4,3)

	1	2	3
1	101	107	109
2	221	231	251
3	311	341	351
4	400	405	477

M(1,1)	101
M(2,1)	221
M(3,1)	311
M(4,1)	400
M(1,2)	107
M(2,2)	231
M(3,2)	341
M(4,2)	405
M(1,3)	109
M(2,3)	251
M(3,3)	351
M(4,3)	477

배열의 주소 계산

	1	2	3
1	101	107	109
2	221	231	251
3	311	341	351
4	400	405	477

◆ 배열(array)

- 2차원 배열의 주소

  - 열 우선방식 Fortran 경우
    m(i, j)의 주소 = 첫데이터주소 + $\{(j-1) * d_1 + (i-1)\} * s$
    s : 데이터 1개가 차지하는 메모리 소자 수

  - 예) Fortran의 경우 m(4,3) 에서
    m(1,1)의 주소 = 80번지에 저장. m(4,2) 의 주소는?
    (단, 정수데이터는 메모리 소자2개 차지)

    $80 + \{(2 - 1) * 4 + (4 - 1)\} * 2 = 94$

확인문제

Fortran 프로그램에서 integer A(5,7) 로 선언된 배열 A의 첫 원소 A(1,1)은 150번지에 저장되어 있으며 integer 하나는 메모리 소자 두 개를 차지한다고 할 때 A(4,6)의 주소는 몇 번지인가?

$$150 + \{(6-1) * 5 + (4-1)\} * 2 = 206$$

포인터의 개념

- 포인터는 메모리 주소를 값으로 가지는 데이터 형(type)이다.
- C 언어에서는 어떤 타입 T에 대해서 T의 포인터 타입이 존재한다.

  int * float *
- 포인터 타입에는 주소연산자(&)와 역참조(간접 지시) 연산자(*)가 사용된다.

[예제1]

```
main()
{
    int *p, q;
    q = 100;
    p = &q;
    printf("%d", *p);
}
```

배열과 포인터

```
float trunc_sum(float *data);

main()
{
    float xarray[10], fsum = 0.0;
    int i;
    printf("Enter 10 reals : ");
    for (i=0; i<10; i++) {
        scanf("%f", xarray+i);
        fsum = fsum + *(xarray + i);
    }
    printf("Sum = %.2f\n", fsum);
    printf("Truncation Value = %.2f\n",
            trunc_sum(xarray));
}
```

- 배열의 이름은 배열 첫 데이터의 주소이고 배열 전체의 대표 정보이다.
- C언어에서는 배열을 함수의 파라미터로 넘겨 줄 때 배열의 이름을 전달한다.

```
for (i=0; i<10; i++) {
    scanf("%f", &xarray[i]);
    fsum = fsum + xarray[i];
}
```

&xarray[0]

배열과 포인터

```
float trunc_sum(float data[])
{
    float sum=0.0;
    int i, ivalue;
    for (i=0; i < 10; i++) {
        ivalue = data[i];
        sum += (data[i]-ivalue);
    }
    return sum;
}
```

(float *data)

실습 예제

◆ [배열을 활용한 프로그래밍 예 1]

다음 프로그래밍을 완성할 수 있도록 함수 avg와 over_avg를 작성하시오.

```
#define DNUM 100
double avg(int a[], int n);
int over_avg(int a[], int n, double average);

main()
{
    int data[DNUM], k, n;
    double aver;
    printf("The number of data : ");
    scanf("%d", &n);
    printf("Enter %d data\n", n);
    for (k=0; k < n; k++)
        scanf("%d", &data[k]);
    aver = avg(data, n);
    printf( “The number of data (over average) : %d\n", over_avg(data, n, aver));
}
```

실습 예제

◆ [배열을 활용한 프로그래밍 예 1]

```
double avg(int a[], int n)
{
    int sum=0, i;
    for (i=0; i < n; i++)
        sum += a[i];
    return (double)sum/n;
}

int over_avg(int a[], int n, double average)
{
    int over=0, i;
    for (i=0; i < n; i++)
        if (a[i] > average) over++;
    return over;
}
```

실습 예제

◆ [배열을 활용한 프로그래밍 예 2]

```
main()
{
    char tname[][10]={"Susan", "Hellen", "Mike"};
    int i, j, ctable[4][3], cnum, total=0;

    printf("This is English Academy.\n There are three teachers : %s   %s   %s\n",
            tname[0], tname[1], tname[2]);
    for (i=0; i < 4; i++) {
       printf("Enter the class size for %d : ", i+1);
       for (j=0; j < 3; j++)
           scanf("%d", &ctable[i][j]);
    }
```

실습 예제

◆ [배열을 활용한 프로그래밍 예 2] (계속)

```
    for (j=0; j < 3; j++) {
        cnum = 0;
        for (i=0; i < 4; i++)
            cnum = cnum + ctable[i][j];
        printf("Student number of %s's class : %d\n", tname[j], cnum);
        total = total + cnum;
    }
    printf( “The number of students in this Academy : %d\n” , total);
}
```

실습 예제

◆ [배열을 활용한 프로그래밍 예 3]

```
main()
{

  char telbook[][2][30] = {
          "Kim kyung sook", "674-1316",
          "Ro mee jin", "2645-1597",
          "Lee woo jin","645-9578",
          "Kim min sook","264-9578",
          "Ko cheol","689-1285",
          "Kang min kook","365-8596",
          "Park seok","369-1258",
          "Do kyung min","659-4859",
          "Rhee Hyunsook", "010-9012-1911",
          "Lee Dawon", "010-212-3737",
          "","",
  };
```

실습 예제

◆ [배열을 활용한 프로그래밍 예 3] (계속)

```
    int i=0, num;

    while (strcmp(telbook[i][0],"")) {
            printf("%d : %s\n", i+1, telbook[i][0]);
            i++;
    }
    printf("================================\n");
    printf("Enter the number(1-%d) : ", i);
    scanf("%d", &num);
    printf("Telephone number of %s : %s\n", telbook[num-1][0], telbook[num-1][1]);

}
```

구조체

레코드

하나의 객체를 활용하기 위하여 정의된 여러 타입의 객체 속성으로 구성된 자료 구조

◆ 구조체의 개념

구조체 : 여러 형의 데이터를 하나의 객체로 선언하여 사용함

레코드 자료구조를 구현함

```
typedef struct member {
        int id;
        char name[20];
        float score;
} example;
```

에 의하여 struct member 라는 구조체 형이 선언됨
동시에 typedef 에 의하여 example이라는
데이터 형을 정의함

```
struct member {
    int id;
    char name[20];
    float score;
};
typedef struct member example;
```

구조체

◆ 구조체의 활용

example onep, exmember[20];

와 같은 선언문에 의하여 구조체를 프로그램에서 변수로 사용할 수 있게 됨.

```
onep.id = 2102;
strcpy(onep.name, “Hong Gildong” );
onep.score = 4.23;
```

// exmember중에 성적이 4.00이상인 학생의 id를 출력하시오.

```
for (k=0; k < 20; k++)
        if (exmember[k].score >= 4.0)
            printf( “%d\n” , exmember[k].id);
```