3. 배열 & 정렬

1. 배열(array)이란?
- 동일한 타입의 데이터가 여러 개 저장되어 있는 데이터 저장장소
- 배열 안에 들어있는 각각의 데이터들은 정수로 되어있는 번호에 의해 접근.
  ↳배열이름+번호 ex) SC[0]

2. 배열의선언
- int scores[10];
  자료형 배열이름 요소의개수(0~9)

인덱스 : 정수상수 or 변수 or 수식

3. 배열과 반복문
- 반복문을 사용해서 배열의요소를 간편하게 처리

```
score[0]=0;
score[1]=0;
score[2]=0;
score[3]=0;
score[4]=0;
```

➡

```
#define SIZE 5
...
for(int i=0;i<SIZE;i++)
    score[i]=0;
```

- 배열정의 ⊕ 반복구조 사용 ⊕ 배열요소의 값들을 난수로 초기화하고 출력

```
#include <stdio.h>
#include <stdlib.h>
#define SIZE 5 #배열크기정의

int main(void)
{
  int scores[SIZE];

  for(int i=0;i<SIZE;i++)
    scores[i]=rand()%100;
  for(int i=0;i<SIZE;i++)
    printf("scores[%d]=%d\n",i,score[i]);
  return 0;
}
```

0~99까지의 난수를 생성하기 위해 난수를 100으로 나눈 나머지값으로 설정

➡실행결과

```
scores[0]=41
scores[1]=67
scores[2]=34
scores[3]=0
scores[4]=69
```

4. 배열의 초기화
- 초기값을 콤마로 분리한 후 중괄호 { }로 감싼다.
  ↳int scores[5]={10,20,30,40,50}
- 초기값을 일부만 주면 나머지는 0으로 초기화
  ↳모든 값을 0으로 초기화시키려면 int scores[5]={0};
- 배열을 지역변수로 선언하면 초기화되지않은 배열은 쓰레기값을 갖게됨.

```
#include<stdio.h>
int main(){
  int arr[]={3,1,4,2...};
  for(int i=0;i<sizeof(arr)/sizeof(int);i++)
  { printf("%d",arr[i]);
  }
  printf("\n");
}
```

int형 하나당 4byte 차지.
배열전체의 크기(4byte x 요소개수)
일일이 요소의 개수를 구하기 위한 방법

5. 배열요소의 개수 계산
- int size = sizeof(scores)/sizeof(scores[0]); ⇒

↳전체배열의크기 ↳배열요소의크기

5. 배열의복사

[잘못된방법]
```
int a[size]={1.2.3.4.5};
int b[size];
b=a;
```

[올바른방법]
```
int i;
int a[SIZE]={1.2,3.4.5}
int b[SIZE];
for(i=0;i<SIZE;i++)
    b[i]=a[i];
```

1. 정렬(sorting)이란?
- 물건을 크기순으로 오름차순 or 내림차순으로 나열하는것

2. 선택정렬(selection sort)이란?

정렬완료된 숫자들의 배열 | 정렬되지않은 숫자들의 배열

가장작은숫자

(선택정렬) 오른쪽 배열이 공백이 될때까지 이 과정을 되풀이하는 정렬기법

3. 탐색이란?
- 탐색의 대상이되는 데이터 : 보통 배열에 저장됨
- 탐색키(search key) : 찾는 특정한 정수
- (순차탐색) 배열의 원소를 순서대로 하나씩 꺼내서 탐색키와 비교하여 원하는 값을 찾아가는 방법

(코드)
```
#include <stdio.h>
#define SIZE 10
int main(void)
{
  int key,i;
  int list[SIZE]={1,2,3,4,5,6,7,8};
  printf("탐색할 값은?");
  scanf("%d",key);

  for(i=0;i<SIZE;i++)
    if(list[i]==key)
      printf("탐색성공인덱스=%d",i);

  printf("탐색종료");
  return 0;
}
```

(코드)
```
#include <stdio.h>
#define SIZE 10
int main(void)
{
  int list[SIZE]={3,2,9,7,1,4,8,0,6,5};

  int i,j,temp,least;

  for(i=0;i<SIZE-1;i++)
  {
    least=i;
    for(j=i+1;j<SIZE;j++)
    {
      least=i;
      for(j=i+1;j<SIZE;j++)
        if(list[j]<list[least])
          least=j;
      temp=list[i];
      list[i]=list[least];
      list[least]=temp;
    }
    for(i=0;i<SIZE;i++)
      printf("%d ",list[i]);
    printf("\n");
    return 0;
}
```

(이진탐색) 속도가 빠른 탐색기법
```
#include <stdio.h>
#define SIZE 16
int binary_search(int list[], int n, int key);

int main(void)
{
  int key;
  int grade[SIZE]={2~47};

  printf("탐색할값?");
  scanf("%d",&key);
  Printf("탐색결과:%d \n", binary_search(grade,SIZE,key));
  return 0;
}

int binary_Search(int list[], int n, int key)
{
  int low,high,middle;

  low=0;
  high=n-1;

  while(low<=high){ //아직 숫자들이 남아있다면
    printf("[%d %d]\n",low,high)//하한,상한 출력
    middle=(low+high)/2; //중간위치 계산
    if(key==list[middle]) //탐색하면 탐색성공
      return middle;
    else if(key>list[middle])//중간원소보다 크면
      low=middle+1; //새로운값으로 low 설정
    else
      high=middle-1;//새로운값으로 high 설정
  }
  return -1;
}
```

4. 이차원배열
- 세로줄 : 행(row) ⊕ 가로줄 : 열(column)
- (3x5) 크기의 정수형 2차원배열 s[]정의 0~99 난수저장후 출력하는 프로그램

↳(코딩)
```
#include <stdio.h>
#include <stdlib.h>
#define ROWS 3 //행개수
#define COLS 5 //열개수
int main(void)
{
  int s[ROWS][COLS]; //2차원배열선언
  int i,j; //2개의인덱스변수

  for(i=0;i<ROWS;i++)
    for(j=0;j<COLS;j++)
      s[i][j]=rand()%100;

  for(i=0;i<ROWS;i++)
  {
    for(j=0;j<COLS;j++)
      printf("%02d ",s[i][j]);
    printf("\n");
  }
  return 0;
}
```

열 / 행

➡출력결과

```
41 67 34 00 69
24 78 58 62 64
25 45 81 29 61
```

5. 이차원배열의 초기화
- int s[3][5]={
  {0,1,2,3,4}, //첫번째행 요소들의 초기값
  {10,11,12,13,14}, //두번째행 요소들의 초기값
  {20,21,22,23,24}, //세번째행 요소들의 초기값
};