컴퓨터프로그래밍 1 복습

첫번째 프로그래밍 수업이었던 컴퓨터 프로그래밍 1에서 우리는 여러 개의 논리를 다루었다.

그 논리들을 다시한번 숙지하고 아래의 논리에 대해서 함수화를 완성해보자.
논리 8: 소수 여부
논리 10: 배열의 동일성 판단
논리 12: 탐색하여 위치 찾기

논리야 놀자 #/ - 반복문의 활용

✪ 논리1: 1부터 n까지의 합을 계산(1 + 2 + 3 + 4 +…+ n)

```
sum = 0; i = 1;
while (i <= n) // n번 반복
{
    sum += i;
    i++;
}
```

✪ 논리2: 1부터 n까지의 곱을 계산(1 * 2 * 3 *…* n)

```
product = 1; i = 1;
while (i <= n) // n번 반복
{
    product *= i;
    i++;
}
```

✪ 논리3: 3을 n번 더함(3 + 3 + 3 + …+ 3)

```
result = 0; i = 1;
while (i <= n) // n번 반복
{
    result += 3;
    i++;
}
```

✪ 논리4: 5을 n번 곱함(5 * 5 * …* 5)

```
result = 1; i = 0;
while (i < n) // n번 반복
{
    result *= 3;
    i++;
}
```

✪ 논리5: n개의 점수를 읽어서 총점을 계산, 평균을 계산(점수1 + 점수2 + …+ 점수n)

```
total = 0; i = 0;
while (i < n) // n번 반복, <= → <
{
    score를 읽는다;
    total 누적;

    i++;
}
```

✪ 논리6: n개의 점수를 읽어서 최대값을 찾는다(점수1, ...점수n중 가장 큰 값)

```
max = 아주 작은값;
i = 0;
while (i < n)
{
    score를 읽는다;
    if (max < score)
        max = score;

    i++;
}
```

논리야 놀자 #2

✪ 논리 7: n의 약수를 찾는다(찾아 출력한다)
약수의 성질은?

```
// 약수를 출력
for (i = 1; i <= n; i++)
   if (n % i == 0)
        printf( "%d₩n" , i);
```

✪ 논리 8: n이 소수인지 아닌지를 판별한다(이다 아니다를 출력)

소수의 성질은?

소수가 아닌걸 발견하면 스톱, *소수라는 판단을 하려면 끝까지 봐야 안다.* prime = 1 //소수라고 일단 설정 i ← 0.. (n-1) if (n이 i로 나누어지면) prime = 0; //소수 아님을 발견!! break; if (prime이 1이면) 소수다 else 소수 아니다	☞ **함수화** //소수인 경우 1을 아닌 경우 0을 반환하는 함수 //int isPrime(int n)을 정의하라 **int isPrime(int n)** { }

논리야 놀자 #3 - 배열(array)을 이용한 논리 연습

✪ 논리 9: 어떤 배열에 대해서 역순 배열 만들기

예: int a[5] = {10, 20, 30, 40, 50};
int r[5];
일 때 a의 원소들을 역순으로 r에 저장해보자.

a[] ← 10 20 30 40 50
r[] ← 50 40 30 20 10

i ← 0..4
    r[?] ← a[i]

✪ 논리10: 두 개의 배열이 같은가를 판별(하여 같다 다르다를 추력)

예: 두 개의 배열 a[], b[]를 비교해보자.

처음에는 일단 같다고 설정.
일단 크기가 다르면, 다르다
크기가 같다면 첫번째, 두번째, ,,,, 마지막 원소를 비교하다가 다른 것이 발견되면, 다르다
다름을 발견하면 스톱,
같다는 판단을 하려면 끝까지(두 배열의 마지막 원소까지 비교..) 해야 안다.

```
same ← 1;
if 크기가 다르면
    same ← 0;
else
    for (I ← 0.. 크기-1)
       다른 것이 발견되면
      same ← 0하고
      for문을 빠져나온다

if same
    같다
else
    다르다
```

☞ **함수화**

```
//두 개의 배열 a, b가 같으면 1을, 같지 않으면 0을 반환하는 함수를
//작성하라. aSize, bSize는 각각 배열 a와 b의 크기이다.

int equalArray(int a[], int aSize, int b[], int bSize)
{




}
```

✪ 논리 11(정렬): 배열의 값을 오름차순(혹은 내림차순)으로 정렬한다.

예: 선택정렬 - 배열 list[] = {5, 3, 8, 1, 2, 7}을 오름차순으로 정렬해보자.

최소값 원소를 첫째 원소랑 바꾸고, 그 다음의 최소값 원소를 둘째 원소랑 바꾸고…

(문제해결기법 수업에서 추후 다룸)

✪ 논리12(탐색): 배열에서 어떤 값(탐색키, search key)을 가진 원소가 있는가를 판별(있다 없다, 혹은 몇 번째에 있다. 여러 개 있을 경우 첫 번째 것으로 한다)

예: a[12] = {11, 22, 33, 44, 55, 66, 11, 22, 33, 44, 55, 66}일 때

33이 배열에 있는가 없는가? 혹은 33이 배열의 몇 번째에 있는가?(배열 안에 33이 여러 개 있을 경우 첫번째것)

key를 발견하면 스톱,
없는 것은 끝까지(배열의 마지막 원소까지)봐야 안다

SIZE를 배열의 크기라고 가정

int a[SIZE] = {11, 22, 33, 44, 55, 66, 11, 22, 33, 44, 55, 66};

//버전1	//버전2	//버전3
keyIndex = -1// 초기값 *i ← 0,…, 9에 대해서 반복* *{* *if (array[i] == 33) // 발견하면* *{* *keyIndex = i;* *break; // 없으면 어떤 일이?* *}* *}* *if (keyIndex == -1) // 없으면* *…* *else // 있으므로*	key를 읽는다 ; found = 0; // 없다고 처음에는 설정 i ← 0..(SIZE-1) if (a[i]이 key이면) found = 1; // 찾았다! break; if (found가 0이면) 없다 else (i+1)번째에 있다	key를 읽는다 ; i ← 0..(SIZE-1) if (a[i]이 key이면) break; if (i == SIZE) 없다 else (i+1)번째에 있다.

☞ **함수화**

//배열 a에 key가 있으면 그 인덱스를, 없으면 -1을 반환하는 함수 search를 작성하라.
//aSize는 각각 배열 a의 크기이다
int search(int a[], int sizeA, int key)
{

}

✪ 논리13: 어떤 특정한 값을 갖는 원소들을 모아 배열에 넣는다.
예: 정수를 10개 읽으면서 홀수이면 배열 odd에 넣고 짝수이면 배열 even에 넣는다.

oddIndex ← 0;
evenIndex ← 0;
10번 반복하다
    수를 읽는다.
    그 수가 홀수이면,
        odd[oddIndex] ← 그 수
        oddIndex++;
    그렇지 않으면(짝수이면)
        even[evenIndex] ← 그 수
        evenIndex++;

논리야 놀자 #4 – 배열(array)을 이용한 논리 연습

✪ 논리 14: 두 수의 값을 바꾼다.

a ← 5, b ← 10일 때
a ←→ b는 어떻게 하나?

```
int a = 5, b = 10;
int temp;

temp = a;
a = b;
b = temp;
```

✪ 논리 15: 배열을 역순배열로 바꾼다

int a[5] = {1, 2, 3, 4, 5}일 때

a[0] ←→ a[4]
a[1] ←→ a[3]
a[2] ←→ a[2] - 할 필요 없음

int a[6] = {1, 2, 3, 4, 5, 6}일 때

a[0] ←→ a[5]
a[1] ←→ a[4]
a[2] ←→ a[3]

```
for (i = 0; i < SIZE / 2; i++)

  a[i] ←→ a[SIZE - i - 1];
```