민 선 생 코 딩 학 원 훈 련 반

수업노트 LV-16

배우는 내용

1. 설계 방법 (배열에서 문자 삽입/삭제)
2. 피보나치 수열 / 배열 누적 값 구하기
3. MAX / MIN 함수 구현
4. Flag를 대신하는 isExist 함수

설계란

- **소스코드를 작성하기 전 미리 계획을 잡는 것**
- 건축을 할 때 미리 설계도를 짜고, 집을 지어야 하는 것
  설계도 없는 프로그램은 코드가 산으로 간다
- 실제로 현업에서 대충 설계를 하거나,
  설계를 잘못된 방향으로 할 경우
  버그가 속출하거나 **처음부터 다시 짜야 하는 경우가 많음**

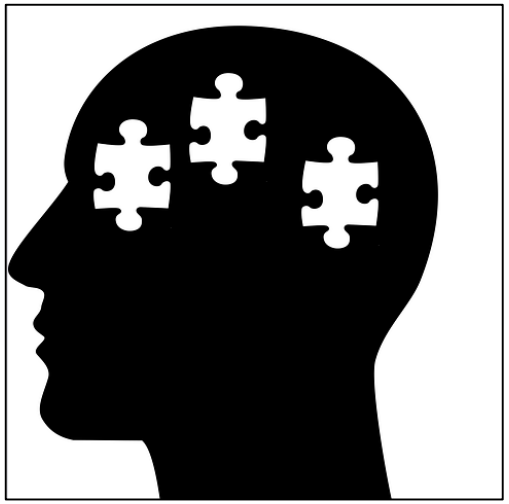
설계는 메모가 필수

- 인간 두뇌의 단기 저장공간이 좁기 때문에,
  참신한 생각의 논리들을 펼칠 두뇌 공간이 부족하다.

  따라서, 주석 or 연습장 에다가 메모를 해가면서 설계도를 만들어야 한다.

- 여러 번 짜 봤던 프로그램이라면?
  주석으로 간단히 설계도 작성

- 프로그램 규모가 크거나 / 처음 짜보는 도전적인 프로그램이라면?
  연습장에 상세히 설계도를 작성



두뇌의 공간이 넓다고 믿지 말자
주석 or 연습장에 메모는 필수

설계방법

- 전체적으로 어떻게 짜야 할 지 적어가면서 고민하는 것
- 두 가지 관점에서 설계를 하면 됨 **(과정 설계 / 세부 설계)**

설계하는 방법1 – 과정설계

- 순차적으로 코딩해야 할 내용들을 계획한다

과정의 예시

① for 5번 돌려 arr 배열에 입력

② 글자수 구하기

③ max값 구하기

④ Min값 구하기

⑤ 출력

배열 중간에 한 글자 집어넣기

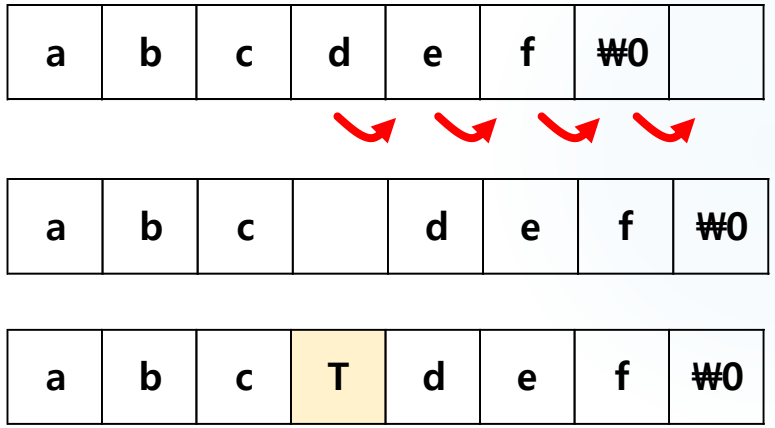
- 한 문장을 입력 받고 'T'라는 글자를 집어 넣으려면?

→ 'd'부터 한 칸씩 각 글자를 뒤로 이동시켜 주어야 함

문장에 문자 넣기 설계

- 연습장 / 주석으로 적으면서 설계하기

1) 문장길이 구하기(NULL문자 발견시, break)
2) 넣을 index부터 한칸씩 뒤로 글자 옮기기
3) 해당 index에 문자 넣기
4) 전체 문장 출력



과정 하나씩 테스트

▶ 설계 한 내용을 하나씩 구현할 때 마다 Test를 할 것

1) 문장길이 구하는 코드 구현하기

→ 문장의 길이가 잘 구해지는지 Ctrl + F10 (Trace)로 Test

2) 넣을 index부터 한칸씩 뒤로 글자 옮겨주기

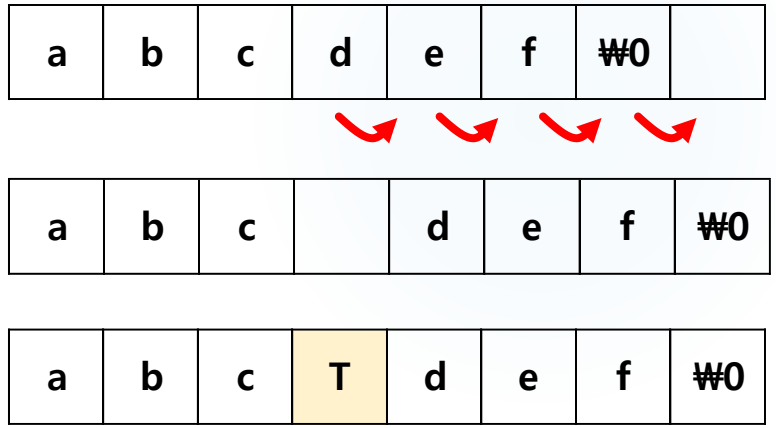
→ 한 칸씩 잘 옮겨지는지 Ctrl + F10 (Trace)로 Test

3) 해당 index에 문자 넣기

→ 예상한대로 잘 넣어지는지 Ctrl + F10으로 Test

4) 전체 문장 출력

→ Ctrl + F10으로 마지막 Test



Test를 많이 / 잘하는 개발자일 수록
보다 안정성있는 프로그램을 만들어 낼 수 있습니다
고수가 되기 위해서는 Test를 많이하는 습관을 기르세요

과정별 주석 남기기

- **한 소스코드 덩어리가 어떤 동작을 하는지 주석으로 남김**
- **주석을 적는 이유**
  1. 해당 코드가 어떤 내용인지 **빠르게 파악하기 위함**
     (나를 위해)
  2. 남을 위해서가 아닌 **나를 위해서** 주석을 적음
     (소스코드를 길게 짜면 작성한 의도를 **잊어버리기 때문**)
- **다른 사람이 주석을 읽어봐도 이해할 수 있도록 적어야 함**

```
char vect[8] = "abcdef";
int index = 3;
int x, len;

//1. 문장의 길이 구하기
for (x = 0; x < 8; x++)
{
    if (vect[x] == NULL)
    {
        len = x;
        break;
    }
}

//2. 한 칸씩 뒤로 옮기기
for (x = len; x >= index; x--)
{
    vect[x + 1] = vect[x];
}

//3. 해당 index에 문자 넣기
vect[index] = 'T';

//4. 출력
cout << vect;
```

버그 찾아내기

- 만약 오른쪽과 같이 x = index ~ len까지 for로 반복시키면서 한 글자씩 옮기면 왜 버그가 발생할까요?

- Trace를 통해 직접 버그의 원인을 찾아보기

```
char vect[8] = "abcdef";
int index = 3;
int x, len;

//1. 문장의 길이 구하기
for (x = 0; x < 8; x++)
{
    if (vect[x] == NULL)
    {
        len = x;
        break;
    }
}

//2. 한 칸씩 뒤로 옮기기
for (x = index; x < len; x++)
{
    vect[x + 1] = vect[x];
}

//3. 해당 index에 문자 넣기
vect[index] = 'T';

//4. 출력
cout << vect;
```

vect[index]에 있는 글자만 계속 복사가 되는 버그 발생

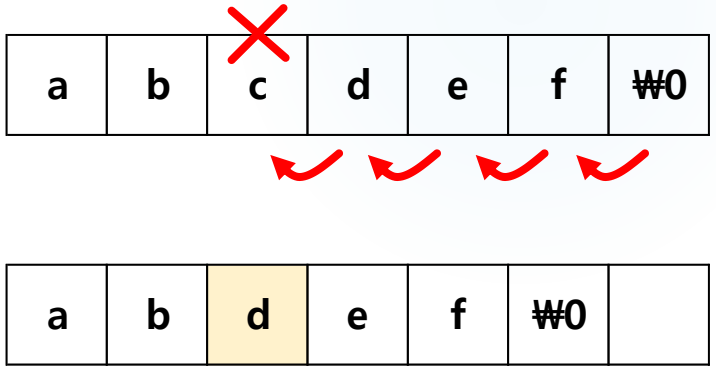
문장에서 한 글자 삭제하기

- 한 문장을 입력 받고 'c'라는 글자를 삭제하기

→ 'd'부터 한 글자씩 앞으로 당겨주어야 함

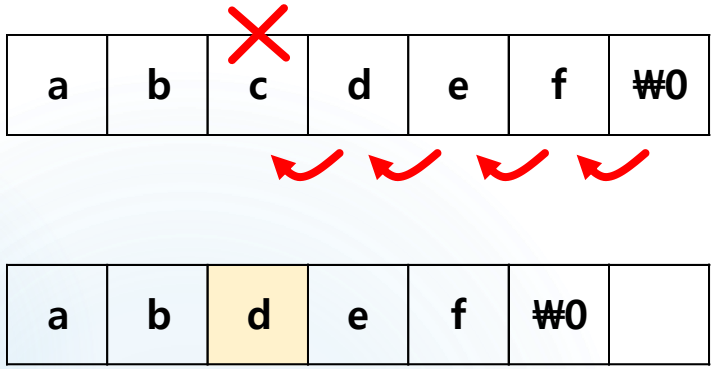
과정 설계하기

1. 문장길이 구하기(주의: NULL문자 발견시, break;)
2. 해당 인덱스까지 문자를 하나씩 당겨오기
3. 전체 문장 출력



문장에서 문자 삭제

- “abcdef” 문장의 2번 index 문자 삭제하고 출력



- 설계 순서

1) 문장길이 구하기(주의: NULL문자 발견시, break;)

2) 해당 인덱스까지 문자를 하나씩 당겨오기

3) 전체 문장 출력

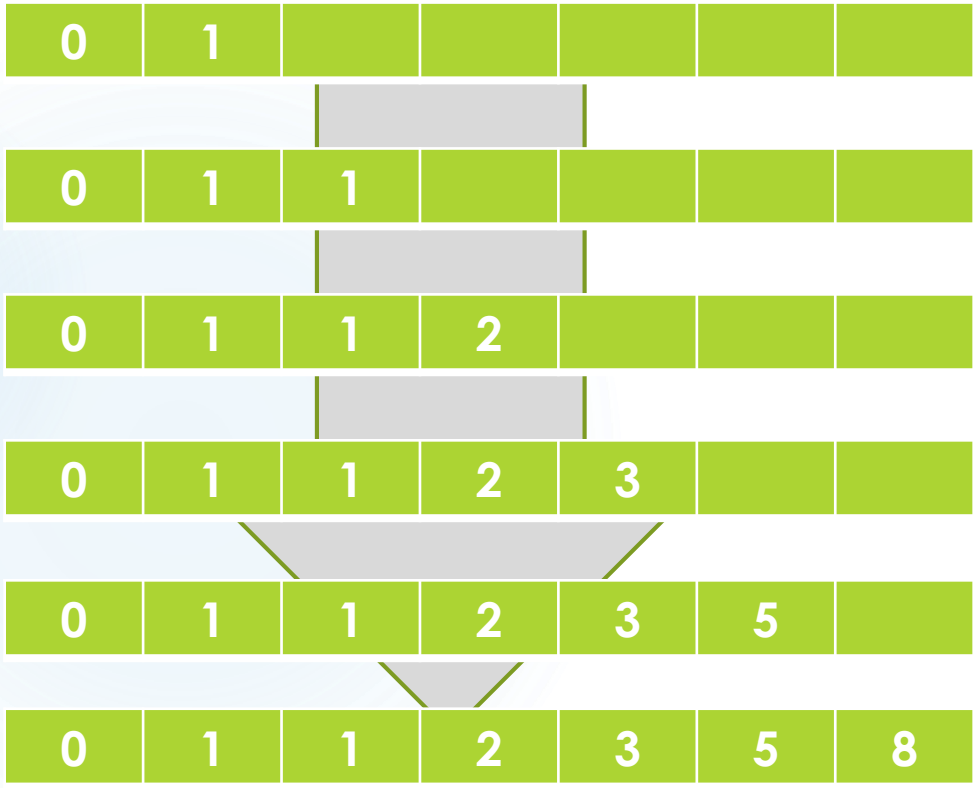
```
char vect[7] = “abcdef";
int index = 2;
int x, n;

//1. 문장의 길이 구하기
for (x = 0; x < 7; x++)
{
    if (vect[x] == NULL)
    {
        n = x;
    }
}

//2. 문자를 하나씩 당겨오기
for (x=index; x < n; x++)
{
    vect[x] = vect[x + 1];
}

//3. 출력
cout << vect;
```

설계하는 방법2 – 세부 과정 설계

- **이 부분에 대해 익숙하지 않은 코딩이라면?**
  → 이 과정에 대한 **세부 설계가 필요**
- **만약 복잡한 for문을 짜야 한다면?**
  - 몇 중 for로 해야 할지 고민
  - **조건 문이 어떤 것이 들어가야 할지 고민**
  - **핵심 코드를 1~2줄 적어보기**
  - 생각한 방법을 **시뮬레이션 해보기**

피보나치 수열

- for문을 연습하기 좋은 문제
- 앞에 두칸의 합을 현재 칸에 넣는 수열




```
int vect[7] = { 0, 1};

for (int x = 2; x < 7; x++)
{
    vect[x] = vect[x - 2] + vect[x - 1];
}
```

MAX함수 만들기

- 최대값의 index를 리턴해주는 문제
- 왜 오른쪽 코드에서 버그가 발생할까요?

```cpp
#include <iostream>
using namespace std;

int vect[5] = { 9, 4, 1, 2, 6 };

int max()
{
    int max = vect[0];
    int maxIndex;

    int x;
    for (x = 0; x < 5; x++)
    {
        if (vect[x] > max)
        {
            max = vect[x];
            maxIndex = x;
        }
    }

    return maxIndex;
}

int main()
{
    cout << max();

    return 0;
}
```

MAX함수 만들기

```cpp
#include <iostream>
using namespace std;

int vect[5] = { 9, 4, 1, 2, 6 };

int max()
{
    int max = vect[0];
    int maxIndex = 0;

    int x;
    for (x = 0; x < 5; x++)
    {
        if (vect[x] > max)
        {
            max = vect[x];
            maxIndex = x;
        }
    }

    return maxIndex;
}

int main()
{
    cout << max();

    return 0;
}
```

- max의 기본값을 vect[0]으로 해주었으면
  maxIndex도 같이 0으로 초기화를 해 주어야 함

- maxIndex를 쓰레기 값으로 두면
  vect[0]이 max값이기 때문에 Runtime Error 발생

Flag 대신 isExist( ) 함수 사용하기1

- 시작반때 배웠던 존재여부를 판단하는 Flag 소스코드
- 이 소스코드를 함수로 빼면
  더 간단하게 코딩할 수 있다

```
#include <iostream>
using namespace std;

int vect[5] = { 9, 4, 1, 2, 6 };

int main( )
{
    int flag = 0;
    int target = 1;

    for (int x = 0; x < 5; x++)
    {
        if (vect[x] == target)
        {
            flag = 1;
            break;
        }
    }

    if (flag == 1) cout << "target 발견";
    else cout << "미발견";

    return 0;
}
```

Flag 대신 isExist( ) 함수 사용하기2

- isExist함수에서 target 값을 찾았으면 1 리턴,
  못찾았다면 0을 리턴하는 함수

- flag 코딩방법과
  isExist 함수를 사용한 코딩방법
  두 방법 모두 숙달 되어야 한다.

```cpp
#include <iostream>
using namespace std;

int vect[5] = { 9, 4, 1, 2, 6 };

int main()
{
    int flag = 0;
    int target = 1;

    for (int x = 0; x < 5; x++)
    {
        if (vect[x] == target)
        {
            flag = 1;
            break;
        }
    }

    if (flag == 1) cout << "target 발견";
    else cout << "미발견";

    return 0;
}
```

```cpp
#include <iostream>
using namespace std;

int vect[5] = { 9, 4, 1, 2, 6 };

int isExist(int target)
{
    for (int x = 0; x < 5; x++)
    {
        if (vect[x] == target)
        {
            return 1;
        }
    }
    return 0;
}

int main()
{
    int ret = isExist(1);

    if (ret == 1) cout << "target 발견";
    else cout << "미발견";

    return 0;
}
```