해커

2022년 2학기
객체지향프로그래밍 교육

정지안(교육부장)

실습: String 자료형 만들기

C/C++에서 문자열은 char 타입의 배열

배열은 구조적으로 단점이 많음 (비효율적)

정적인 크기를 가짐

실제 데이터는 문자열 형태가 많음

➔ 개선 필요

실습: String 자료형 만들기

기존 배열형 문자열의 문제점

1. 정적인 크기
2. 문자열 간 연산 불가능
3. Replace, Sort 등 메소드 부재

실습: String 자료형 만들기

Python String 타입

1. 동적인 크기(크기 지정 안함)
2. +, * 연산 가능
3. 다양한 메소드 존재

➔ 목표 : Python의 String 타입 C++로 구현

실습: String 자료형 만들기 - 1. 구조 파악

W3School Python docs:
https://www.w3schools.com/python/python_strings.asp

Minor 기능, 구현이 어려운 기능 ➔ 구현 X

필요한 기능만 골라 구현

실습: String 자료형 만들기 - 1. 구조 파악

1. 내부적으로 배열형 문자열 저장

2. 문자열 메소드 보유

3. 문자열 연산자 보유

4. 동적 크기를 가지는 문자열 구현

실습: String 자료형 만들기 - 2. 클래스 구조 설계

클래스 이름(자료형 이름) : String

멤버)
private:
    문자열 변수 (char 배열)
    length (배열 길이, int)
public:
    생성자()
    소멸자()
    length()
    append(char)
    find(char)
    replace(char, char)
    print()

실습: String 자료형 만들기 - 2. 클래스 구조 설계

```
class String{
private:
    char* str;
    int len;
    int maxSize;
public:
    String(); // default 생성자
    String(int); // 최대 길이 지정 생성자
    ~String(); // 소멸자
    int length(); // 문자열 길이
    void append(char); // 문자열 맨 뒤에 char 변수 추가
    bool find(char); // 문자열 내 특정 char 변수 존재 여부 검사
    void replace(char, char); // 문자열 내 특정 인덱스 대체
    void makeEmpty(); // 문자열 비우기(초기화)
    void print(); // 문자열 출력
};
```

실습: String 자료형 만들기 – 3. 구현

생성자

```
String::String() {
    str = new char[100];
    len = 0;
}

String::String(int max) {
    str = new char[max];
    len = 0;
}
```

실습: String 자료형 만들기 - 3. 구현

소멸자

```
String::~String() {
    delete[] str;
}
```

실습: String 자료형 만들기 - 3. 구현

length()

```
int String::length() {
    return len;
}
```

실습: String 자료형 만들기 - 3. 구현

append() - 직접 구현

실습: String 자료형 만들기 – 3. 구현

find()

```
bool String::find(char x) {
    for (int i = 0; i < len; i++) {
        if (str[i] == x) {
            return true;
        }
    }
    return false;
}
```

실습: String 자료형 만들기

replace() - 직접 구현

실습: String 자료형 만들기

makeEmpty() - 직접 구현

실습: String 자료형 만들기

print() – 직접 구현