과제 063 - My String 클래스

제출 마감일: 2023-04-28 23:59
업로드 가능한 파일 수: 2
제출 방식: 개인

주의

과제 063 은 keylog 분석 및 jplag 를 이용한 표절 검사가 상시 수행됩니다.

- 키로그 파일(p63.csv)에 소스 코드 이름이 제출 파일명과 일치하지 않으면 로그 점수가 계산되지 않습니다.
- 키로그 파일(p63.csv)에 키보드 입력 정보가 없는 경우 로그 점수가 계산되지 않습니다. (로그가 IdeState 나 Action 타입 밖에 없는 경우가 자주 있습니다.)
- 구현이 작은 소스 코드는 주어진 코드를 복사하지 말고 직접 타이핑 하는 것을 권장 드립니다.
- 계속 잘 안될 때에는 초심으로 돌아가 activitytracker 를 clear 하고, 소스 파일명을 제출 파일명과 일치 시킨 후 직접 키보드로 코딩하는 것을 추천 드립니다.

문제

주어진 String 클래스 선언부를 참고하여 String 클래스를 구현하시오.
단, default operators 중 생성자, 복사 생성자, 소멸자를 구현하시오.

<참고>

- String.h, main.cpp 파일은 수정하지 마시오.
- String.cpp 에서는 기본 생성자, 생성자, 복사 생성자, empty(), size(), append() 함수를 구현하시오.
- 주어진 main 의 test cases 가 정상적으로 동작하는지 확인하세요.

// String.h

```
#include <iostream>
#include <algorithm>

class String {
public:
   String() ;
   explicit String(const char* s) ;
   ~String() noexcept ;
   String(const String& s) noexcept ;
```

```
public:
    const char* data() const;
    bool empty() const;
    size_t size() const;
    String& append(const String& str);
    String& append(const char* str);


private:
    static int count_;
    char* data_;
    size_t len_;
};



// main.cpp

#include <iostream>
#include <cassert>
#include <cstring>
#include "String.h"

void test_case1();
void test_case2();
void test_case3();
void test_case4();

int main(){
    std::cout <<  "\n\n" << "Testing the default constructor: " << "\n\n";
    test_case1();

    std::cout <<  "\n\n" <<  "Testing the constructor that takes a C-style string:" << "\n\n";
    test_case2();

    std::cout <<  "\n\n" << "Testing the copy constructor:" <<  "\n\n";
    test_case3();

    std::cout <<  "\n\n" << "Testing the append() function:" <<  "\n\n";
    test_case4();
}

void test_case4() {
    String s11("hello");
    String s12("world");
    s11.append(s12);
    assert(!s11.empty());
    assert(s11.size() == 10);
    assert(strcmp(s11.data(), "helloworld") == 0);

    String s13("");
    String s14("test");
    s13.append(s14);
    assert(!s13.empty());
    assert(s13.size() == 4);
    assert(strcmp(s13.data(), "test") == 0);
}

void test_case3() {
    String s3("hello");
    String s4(s3);
    assert(!s4.empty());
```

```
    assert(s4.size() == 5);
    assert(strcmp(s4.data(), "hello") == 0);

    String s5("");
    String s6(s5);
    assert(s6.empty());
    assert(s6.size() == 0);
    assert(s6.data()[0] == '\0');
}

void test_case2() {
    String s1("hello");
    assert(!s1.empty());
    assert(s1.size() == 5);
    assert(strcmp(s1.data(), "hello") == 0);

    String s2("");
    assert(s2.empty());
    assert(s2.size() == 0);
    assert(s2.data()[0] == '\0');
}

void test_case1() {
    String s;
    assert(s.empty());
    assert(s.size() == 0);
    assert(s.data()[0] == '\0');
}
```

입력

없음

출력

```
Testing the default constructor:

Default Constructor
# of String object: 1
Destructor
# of String object: 0


Testing the constructor that takes a C-style string:

Constructor
# of String object: 1
Constructor
# of String object: 2
Destructor
# of String object: 1
Destructor
# of String object: 0
```

Testing the copy constructor:

```
Constructor
# of String object: 1
Copy Constructor
# of String object: 2
Constructor
# of String object: 3
Copy Constructor
# of String object: 4
Destructor
# of String object: 3
Destructor
# of String object: 2
Destructor
# of String object: 1
Destructor
# of String object: 0
```

Testing the append() function:

```
Constructor
# of String object: 1
Constructor
# of String object: 2
Constructor
# of String object: 3
Constructor
# of String object: 4
Destructor
# of String object: 3
Destructor
# of String object: 2
Destructor
# of String object: 1
Destructor
# of String object: 0
```

제출파일

String.cpp
p63.csv