PIC 10A: HOMEWORK 6

WONJUN LEE
UCLA MATHEMATICS DEPARTMENT

Important Notes: Please follows these items to avoid penalties.

(1) Name the files as instructed in the document. If you name them incorrectly, you will receive **0 pts.**

(2) Make sure your codes are successfully compiled and run on Visual Studio on Windows. If your codes do not build on Visual Studio, you will receive **0 pts.**

(3) Submit your homework by 11:59pm on Thursday **(August 4, 2022)**. If you submit it between 12:00am on Thursday and 11:59pm on Friday, your late penalty will be **-40%**. If you submit after 12:00am on Saturday, you will receive **0 pts.**

Date: July 26, 2022.

Problem 1

A template to answer this homework is provided on Burinlearn. It contains three files:

- `main.cpp` contains some demonstration code. You can edit this file as much as you like, and you should edit this file in order to test the functionality of the new class that you create.

- `MyString.hpp` and `MyString.cpp` are the files which you will turn in.

- `MyString.hpp`: This contains the definition of the class `MyString`.

  The beginning says

```
public:
  MyString()       : s(0)   {}
  MyString(char c) : s(1,c) {}

  MyString substr(size_t pos, size_t len = -1) const;

  size_t  find(char c, size_t pos =  0) const;
  size_t rfind(char c, size_t pos = -1) const;

  size_t  find(const MyString& str, size_t pos =  0) const;
  size_t rfind(const MyString& str, size_t pos = -1) const;

private:
  std::vector<char> s;
```

  We'll store the string as a `vector` of `char`s which we keep `private`.
  You can see a constructor with no parameters which creates the empty `MyString`.
  You can see a constructor with one parameter which creates a one character `MyString`.
  I have provided declarations of `substr`, and overloaded `find` and `rfind`. You need to write proper declarations for the other functions.

- `MyString.cpp`: This contains incomplete definitions of `substr`, `find`, and `rfind`.

Here are your tasks...

(1) Check this link to see the description of the functions that you will write for this homework.
(2) Define a new constructor that has two parameters: `size_t n`, `char c`. It should create a new `MyString` with `n` characters all equal to `c`.
(3) Define member functions `length, empty, push_back, pop_back` just like for `string`.
(4) Define an overloaded member function resize (each definition is still just one line).

```
void resize (size_t n);
void resize (size_t n, char c);
```

This should resize the MyString to a length of n characters.
If n is smaller than the current length, the MyString should be shortened to its first n characters, removing the characters beyond the nth.
If n is greater than the current length, the MyString should be extended by inserting at the end as many characters as needed to reach a size of n.

If c is specified, the new elements should be initialized as copies of c, otherwise, they take on the value '\0' (the null character).

(5) The function definitions of `length, empty, push_back, pop_back, resize` are very short. Make these definitions in the class interface. Each of mine is one line long. **Remember that your member variable s has all of the member functions of `vector<char>` available to it.**

(6) In `MyString.cpp`, give appropriate definitions for `substr`, and overloaded `find` and `rfind`.

(7) The first declaration of find reads

```
size_t find(char c, size_t pos = 0) const;
```

This means that if `str` is an instance of `MyString`, calling `str.find('!')` is the same as calling `str.find('!',0)`. When `pos` is not specified it takes on the default value of `0`. You should write your definition to work for any value of `pos`.

(8) Notice that a `size_t` can never be negative. When I `return -1` in the "empty" definitions, that casts to a `size_t`, and `static_cast<size_t>(-1)` is the biggest `size_t` there is: $2^{64}-1$ on Mac machines. Don't mess up by writing a while loop for `rfind` that says something like `while(pos >= 0)`. Such a while loop would go on forever. Also, don't hack your way around this by converting to `int`s: that loses information and could create another bug. Instead use a while loop like `while(pos != -1)`. In order to make this comparison, `-1` is correctly cast as a `size_t`.

(9) Because `size_t` cannot be negative you should be careful about subtracting numbers. In `substr` you may want to check whether `pos < s.size()` before performing the subtraction `s.size()-pos`.

(10) Finally, here is a new `main.cpp` to test `substr`, `find`, and `rfind`.