Lab3. String

For this problem, we hope all of you can learn how to do the dynamic memory allocation. We will provide you template class file, you have to complete class methods by yourself . For this lab section, we are going to design a class of string, the SPEC. of the class is shown below (the class name is MyString).
After you complete all the methods of the class, you should ask the TA for the DEMO.

We provide 3 files for you and you should link these files together on your IDE:
- MyString.h
- MyString.cpp
- main.cpp

Don't change anything in MyString.h and main.cpp. You need to implement the methods in MyString.cpp.

Class member :

There are two private members in the class:
- char * _s: char array, which is used to store the string.
- int _size: Represent the size of string(_s).

Constructors & Destructor:

- MyString() :
  The default constructor of the class, the default size of the string is 0, and the string(_s) point to null.
- MyString (const string &s) :
  Construct the class, and the string(_s) value should be equal to parameter s.
- ~ MyString ():
  The default destructor, free all the memory you have used for this class.

Class methods :

- size () :
  return the length of string.
- at (const int &index) const : return the specified location(parameter index) of character, if index is out of range, just print "Error in at()" on the screen and return '\0'.
- replace (const int &index, const char &ch) : replace the specified location(parameter index) of character to new one(parameter ch), if index is out of range, print "Error in replace()".
- set (const std::string &s) : set your string(_s) to the new string(parameter s).
- append (const std::string &s) : append a new string(parameter s) to your string(_s)
- reverse () : Reverse the string, you have to complete this method by yourself, don't call any other function like std::reverse().
- print () : Print the string on screen.
- clear () : Clear the string, set size to zero and deallocate the string.

Example :

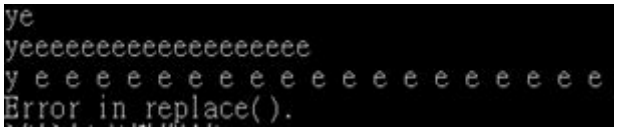
If we call the following code :

```
MyString str("ye");
str.print();
str.append("eeeeeeeeeeeeeeeeee");
str.print();

for (int i = 0; i < str.size(); i++) {
        cout << str.at(i) << " ";
}
cout << endl;

str.replace(100, 's');
```

The result should be like this :



※The main.cpp that we provided is not this one.

Hint :

1. In C, if we use char array to represent the string, we always put ‘\0’ in the end of the string.
2. When you try to allocate the memory, you have to allocate the proper size to your string.
3. You have to be careful when using the pointer, for example : you can’t do anything to the null pointer, or your program will be crash.
4. You need to deallocate all the memory that are not being used.
   (沒用到的記憶體記得清掉)