Lab 4

Object Oriented Programming

Instructions:

- **Attempt the following tasks exactly in the given order.**
- You are required to create a **multi-file project** for each task.
- Indent your code properly.
- Use meaningful variable and function names. Follow the naming conventions.
- Use meaningful prompt lines and labels for all input/output.
- Make sure that there are **NO dangling pointers** or **memory leaks** in your program.

Objective:

- As discussed earlier that you are weak in string manipulation: So in this lab we shall make replica of CString library available in Visual C++. Do it with full focus because this lab will be used heavily in upcoming labs and assignments.

CString ADT

You have to define an Abstract Data Type (ADT) named CString having following data structure:

```
char * str;        // pointer to an array of characters
int size;          // size of the array (it will always represent amount of
                   memory allocated to str pointer)
```

Methods

1. CString ( ) ;
Default constructor

2. CString (char c) ;
Create an object with one character

3. CString(char*) ;
Conversion constructor

4. CString (const CString &);
Copy constructor

5. ~CString ( ) ;
Free the resources if any captured by calling object.

6. int getLenght ( )
Return Value
Return the length of string

7. void display( ) const ;

Display on console the string contained in calling object.

8. int find(char* substr , int start=0) const ;

SubStr

A substring to search for.

Start

The index of the character in the string to begin the search with, or 0 to start from the beginning

Return Value

The first character index of the string that matches the substring, -1 otherwise

9. int find(char ch, int star t=0) const ;

ch

A character to search for

Start

The index of the character in the string to begin the search with, or 0 to start from the beginning

Return Value

The character index of the string that matches the ch, -1 otherwise

10. int insert (int index, char* substr) ;

Index

The index of the character before which the insertion will take place (start to insert at index and move older elements ahead)

substr

A pointer to the sub-string to be inserted

Return Value

The length of the changed string

11. int insert (int index, char ch) ;

Index

The index of the character before which the insertion will take place (start to insert at index and move older elements one index ahead)

ch

A character to be inserted

Return Value

The length of the changed string

12. CString left (int count) const ;

Count

The number of characters to extract from calling CString object from left side

Return Value

A CString object that contains a copy of the specified range of characters

13. CString right (int count) const ;

Count

The number of characters to extract from calling CString object from right side

Return Value

A CString object that contains a copy of the specified range of characters

14. int remove(int index, int count=1) ;

Index

The zero-based index of the first character in the CString object to delete.

Count

The number of characters to be removed

Return Value

The length of the changed string

15. int remove (char ch) ;

ch

The character to be removed from a string

Return Value

The count of characters removed from the string. Zero if the string is not changed

16. void replace(char new) ;

New

Character replacing all the old characters in old string : string is filled with new char

17. int replace(char old, char new) ;

Old

The character to be replaced by n*ew*

New

Character replacing *Old*

Return Value

Return the number of times the old char is replaced by new char

18. int replace(char* old, char* new) ;

Old

A pointer to a string containing the character to be replaced by n*ew*

New

A pointer to a string containing the character replacing *Old*

Return Value

Return the number of times the old string is replaced by new string

19. void trim( ) ;

Removes all leading and trailing occurrences of white spaces

20. void trimLeft();

Removes all white spaces on left side of calling object

21. void trimRight ();
Removes all white spaces on right side of calling object

22. void makeUpper ();
change the string characters to upper case

23. void makeLower ( ) ;
change the string characters to lowercase

24. void reverse( ) ;
reverse the order of character in old string i.e. reverse them.

25. void Resize(int add) ;
This function will take an integer variable and increase the string buffer of the object and also preserving the old data.

26. CString concat (const CString& *s2*) const ;
Concatenate the calling and received object
Return Value
Return value concatenated object

27. void concatEqual (const CString& *s2*);
Concatenate the calling and received object

28. void concatEqual (char * s2);
Concatenate the calling and received string.

29. int isEqual (const CString & s2) const ;
compares the calling and received object
Return Value
follow the strcmp in the string.h criteria for these comparisons

30. int isEqual (char * s2) const ;
Compares the calling and received string
Return Value
follow the strcmp in the string.h criteria for these comparisons

31. void input ( ) ;
Takes input from console in calling object.

32. char & at (int index) ;
index
Receives the index for string.
Return Value
reference of array location represented by index

33. int isEmpty( ) ;
Tells whether string is empty or not
Return Value
return 0 if string empty and nonzero value if string is not empty.