University of Central Punjab
(Incorporated by Ordinance No. XXIV of 2002 promulgated by Government of the Punjab)
FACULTY OF INFORMATION TECHNOLOGY

# Object Oriented Programming Section C15

## Session: Fall 2022

| Assignment no 4: handwritten Due on Two Parts:<br>PART ONE on Tue Dec 27, 2022 in Class Total Marks = 50<br>PART TWO on Tue Jan 03, 2022 in Class Total Marks = 100<br>Each Delay of a day will cost 20 % of Marks | |
|---|---|
| **Topic** | Composition |
| **Objective/ Outcome** | Making students familiarize with Composition and its benefits. |

**Instructions:** **(20 % marks will be deducted from the total obtained if these instructions are not followed)**

1. Indent your code.
2. Comment your code.
3. Use meaningful variable names.
4. Plan your code carefully on a piece of paper before you implement it.
5. Name of the program should be same as the task name. i.e. the first program should be task_1.cpp
6. **Do all the tasks in multiple files (separate interface and implementation).**
7. **void main() is not allowed. Use int main()**
8. **You are not allowed to use system("pause")**
9. **You are not allowed to use any built-in functions**
10. **Implement it in multiple files. Every task will contain three respective files Class.h (declarations) Class.cpp (definitions) and task.cpp (main().cpp)**

**You are required to follow the naming conventions as follow:**

11. **Class Name**: Should be noun and First Letter Capital (use Mixed case for more than one word) and .h and .cpp files must have Class Name as prefix. E.g. Class ComplexNumber must be declared in ComplexNumber.h and defined in ComplexNumber.cpp
12. **Variables:** firstName; (no underscores allowed)

13. **Function:** getName(); (no underscores allowed)
14. **ClassName:** BankAccount (no underscores allowed)
15. **Constant members all UPPERCASE:** MAX_LIMIT
16. **All Classes must contain the following:**
    1. Default, Parameterized and Copy Constructors **(use deep copy where required) [use of const and all parameters must be passed by reference]**
    2. Getter/Setter for all private variable **(use deep copy where required) [use of const and all parameters must be passed by reference]**
    3. Destructor
    4. Class Operators: =, == **(use deep copy where required) [use of const and all parameters must be passed by reference]**
    5. Non-Class Operators << and >> **(use of friend is not allowed, zero marks if used)**

## PART ONE Declare and define the following classes (BUILT FUNCTION ARE NOT ALLOWED e.g. strcpy, strlen, strcmp etc.)

## Due in Class Tue Dec 27, 2022 Total Marks = (25 + 25 = 50 Marks)

(**hint: you may reuse the Code of MyString of the Assignment 1 or LAB 9)**

**FOR ALL DYNAMIC ARRAYS YOU WILL NEED TO PROVIDE THE FOLLOWING NON-CLASS FUNCTIONS FOR ALL THE CLASSES BELOW:**

- **growArray that grows the array by a given value**
- **shrinkArray that shrinks the array by a given value**
- **deepCopy that copies a given array to another array**

**THE OPERATORS << AND >> SHOULD BE ABLE TO WRITE TO AND READ FROM TEXT FILE**

### Class One: Word [25 marks]

**Private Variables:**

a. Cstring

**Public Functions:** (Other than the above Requirements for all Classes)

**(Other than the above Requirements for all Classes)**

i. **Find** that finds if a given character is in the Cstring
ii. **Find** that finds if a given Cstring is present (substring) in the Cstring
iii. **count** that counts how many times a given charcater is present in the Word
iv. **Length** that return the number of character is in the Cstring

v. **Compare** that compares a given argument Word with the current Word, returns 0 if equal, -1 id less and 1 if greater
vi. **toLowercase** that coverts the all characters if the Word to lowercase
vii. **toUppercase** that coverts the all characters if the Word to uppercase

**Class Operaor:**

i. += that appends given argument
ii. < and >

**Non-Class (Global) Operaor:**

+ that concatenate two Words to produce another resultant Word

## Class Two: Sentence [25 marks]

**Private Attributes:**

i. Dynamic Array of "Word" (class one)
ii. Size of the Dynamic Array

**Public Functions: (must use deep copy where required)**

**(Other than the above Requirements for all Classes)**

i. **toLowercase** that coverts the all characters of Each Word of the Array to lowercase
ii. **toUppercase** that coverts the all characters of Each Word of the Array to uppercase
iii. **find** that finds if a given "Word" is present in the Sentence
iv. **count** that counts how many times a given charcater is present in the sentence

**Operators:**

i. Comparison: < and >
ii. += that adds the Array of the Argument to the Array of the Class **(there should be only one fullstop in the result)**

**Global (free or non-class operators):**

+ that adds arrays of two Argument Sentences and return the combined sentence. **(there should be only one fullstop in the result)**

# PART TWO Declare and define the following classes

## Due in Class Tue January 03, 2022 Total Marks = (25 + 25 + 25 + 25 = 100 Marks)

## Class Three: Paragraph [25 marks]

**Private Attributes:**

i. Dynamic Array of "Sentence" (class two)
ii. Size of the Dynamic Array

**Public Functions:** (**must use deep copy where required)**
**(Other than the above Requirements for all Classes)**

iii. **toLowercase** that coverts the all characters of Each Sentence of the Array to lowercase
iv. **toUppercase** that coverts the all characters of Each Sentence of the Array to uppercase
v. **find** that finds if a given "Cstring" is present in the Paragraph (any sentence of the Paragraph)
vi. **count** that counts how many times a given charcater is present in the Paragraph
vii. **count** that counts number of charcaters in the Paragraph

**Operators:**

i. Comparison: < and >
ii. += that adds the Array of the Argument to the Array of the Class

**Global (free or non-class operators):**

+ that adds arrays of two Argument Paragraphs and return the combined Paragraph.

## Class Four: Chapter [25 marks]

**Private Attributes:**

i. Dynamic Array of "Paragraph" (class three)
ii. Size of the Dynamic Array

**Public Functions:** (**must use deep copy where required)**
**(Other than the above Requirements for all Classes)**

i. **toLowercase** that coverts the all characters of Each Paragraph of the Array to lowercase
ii. **toUppercase** that coverts the all characters of Each Paragraph of the Array to uppercase
iii. **find** that finds if a given "Cstring" is present in the Chapter (any Paragraph of the Chapter)
iv. **count** that counts how many times a given charcater is present in the Chapter
v. **count** that counts number of charcaters in the Chapter

**Operators:**

i. Comparison: < and >
ii. += that adds the Array of the Argument to the Array of the Class

**Global (free or non-class operators):**

+ that adds arrays of two Argument Chapters and return the combined Chapter.

## Class Five Book : [25 marks]

**Private Attributes:**

i. Dynamic Array of "Chapter" (class four)
ii. Size of the Dynamic Array

**Public Functions:** (**must use deep copy where required)**

**(Other than the above Requirements for all Classes)**

iii. **toLowercase** that coverts the all characters of Each Chapter of the Array to lowercase
iv. **toUppercase** that coverts the all characters of Each Chapter of the Array to uppercase
v. **find** that finds if a given "Cstring" is present in the Book (any Paragraph of the Chapter)
vi. **count** that counts how many times a given charcater is present in the Book
vii. **count** that counts number of charcaters in the Book

**Operators:**

i. Comparison: < and >
ii. += that adds the Array of the Argument to the Array of the Class

**Global (free or non-class operators):**

+ that adds arrays of two Argument Books and return the combined Book.

## Write a Main Function to test the Composition at Work [25 marks]

**You may declare and define global functions that can be used by the following main function**

The main function must do the following:

1. Declare Two variables, bookOne and BookTwo, of Type Book, using Default Constructor
2. For First Variable, use >> operator to read the data of the book from a file named "book_one.txt" (sample file is given at the end of this doc)
3. For Second Variable, use >> operator to read the data of the book from a file named "book_two.txt" (sample file is given at the end of this doc)
4. Use "compare" Non-Class function to display if bookOne is equal, greater or less than book Two
5. Use toUppercase function of the class Book to Convert bookOne to uppercase
6. Use toLowercase function of the class Book to Convert bookTwo to lowercase
7. Use global + operator to combine bookOne and bookTwo into a third bookThree
8. Display count of characters, words, sentences and chapter in bookOne, bookTwo and bookThree
9. Display the count of Vowels in bookOne, bookTwo and bookThree
10. Check and display if the cstring "compromise" is in bookOne, bookTwo and bookThree
11. A text file is given named "dictionary.txt" that contains a list of words, display those words in bookOne that are not in the "dictionary"

12. For each word of the file “dictionary.txt”, display the count for each word the number of times that word present in bookThree