Lab Exam(Group A) **May9th , 2024**

Lab Exam: Multi-dimensional Points - The Swiss Army Knife Class **Marks:15**

Objective:

In graphics, simulation, or game applications, representing multi-dimensional points is a common requirement. However, creating separate classes for points with different dimensions (e.g., Point2D, Point3D, Point4D, etc.) can lead to maintenance challenges due to code duplication. Your task is to design a versatile class template that can handle points of any dimension with components of any fundamental numeric type in C++.

Instructions:

a. Implement a class template named Point that allows setting up points with any dimension.
b. The number of components defaults to 2. In other words, we want to create two dimensional points of non-type template parameter i.e: unsigned integer by default for a code like below:

```
Point point_2d;
point_2d.set_value(0,10);
point_2d.set_value(1,20);
point_2d.print_info() ;
```

should compile, link , run and produce output like below

point : [10 20]

and Code like below

```
Point<double,3> point_3d;
point_3d.set_value(0,10.1);
point_3d.set_value(1,20.2);
point_3d.set_value(2,30.3);
point_3d.print_info() ;
```

should compile, link , run and produce output like below

point : [10.1 20.2 30.3]

c. The class should provide the following functionalities:
   i. set_value: Set the value of a component at a given index.
   ii. get_value: Get the value of a component at a given index.
   iii. print_info: Print the point information in the format specified below.
   iv. Ensure that the class template can handle points with components of any supported fundamental numeric type in C++.
   v. write an output stream insertion operator, in such a way that we can use our point class like below

```
Point<int,3> point_3d;
```

point_3d.set_value(0,10);

point_3d.set_value(1,20);

point_3d.set_value(2,30);

std::cout << point_3d << std::endl;

and this should print something like

Point : [dimension : 3, components : 10 20 30]

d. Example Usage:

Point<int, 3> point_3d;

point_3d.set_value(0, 10);

point_3d.set_value(1, 20);

point_3d.set_value(2, 30);

point_3d.print_info(); **// Output: point : [10 20 30]**

Point<double, 2> point_2d;

point_2d.set_value(0, 10.22);

point_2d.set_value(1, 20.11);

point_2d.print_info(); **// Output: point : [10.22 20.11]**

double value = point_2d.get_value(0); **// value = 10.22**

e. Requirements:

i. Ensure the formatting of the output from print_info() method adheres to the specified format.
ii. Handle out-of-bounds index access gracefully in set_value() and get_value() methods.
iii. Submit the PointTemplate.cpp source file.

f. Evaluation Criteria:

i. Correctness of the implemented functionalities.
ii. Clarity and readability of the code.

Task2: Implementing a smart pointer **15marks**

Write a template class `SmartPointer` that mimics the behavior of a smart pointer. The class should be able to hold a pointer of any type and should release the memory when it is no longer in use (i.e., implement basic reference counting).

What skills this question evaluates: This question assesses the understanding of dynamic memory management and the concept of smart pointers in C++, particularly the implementation of reference counting to manage memory automatically.

Task3: Language Checker using STL Stack **10marks**

Objective: Write a C++ function that uses a stack to determine whether a string is in the language L,

where L = {s: s contains equal numbers of A's and B's}.

Function Signature:

bool isInLanguage(const std::string& str);

Instructions:

A. Implement a function named **isInLanguage** that takes a string as input and returns true if the input string belongs to the language L, and false otherwise. Use the STL stack class to keep track of the number of 'A's and B's encountered.

Write a main() program to tett the above functions.

Test function with following test cases.

1) Input: "ABABAB"

   Output: true

Explanation: The string "ABABAB" belongs to the language L as it has an equal number of 'A's and 'B's.

2) Input: "AAABBBB"
   Output: false

Explanation: The string "AAABBBB" does not belong to the language L as it has more 'B's than 'A's.