**Take classes from Lab 10 task 1.**

**Task 1:**

Write a template function ***sum*** with 2 parameters, first is template type array and second is size of

array (int), return type is also same type of template. Test your function by writing given code in main function. First run with commented code than complete Complex class given on right hand side and test that your code is working for Complex class as well. Take care for all the required functions in Complex class. Write your template function as you write for integer array. Set precision to 2 in overloading of stream insertion operator for Complex class.

|  |  |
| --- | --- |
| **//code of main function**  srand(time(0));  int x[]={3,2,4,6};  double d[]={2.3,4.1,5.2};  cout<<sum(x,4)<<"\n";  cout<<sum(d,3)<<"\n";  Complex cArr[3];  int i;  for (i=0;i<3;i++ )  cout<<cArr[i];  cout<<"\n";  cout<<sum(cArr,3); | **//Incomplete code of Complex class**  #define RND1 rand()%10-5  #define RND2 (rand()%10+1)  class Complex  {  double real, comp;  public:  Complex(){  real=RND1+1.0/RND2;  comp=RND1+1.0/RND2;  }  }; |

**Task 2: (Set class)**

Create a template class Set where Set has data members a generic type dynamic array, current size

(int) & size (int). Set class should be flexible to add any number of elements, means if we add more & more elements, size should increase dynamically. Write parameterized constructors:

* *Single parameter size declare array only*
* *Two parameters template type array and size, declare array according to size & copy elements*

Add member functions add (unique elements).

Overload minus "-" operator such that:

* ***S1 – S2: is a new Set with all the elements of S1, which does not exist in S2.***

Overload plus "+" operator such that:

* ***S1 += S2: Add all unique elements of S2 in S1.***

Implement stream insertion operator. Write a main function to test Set class. Test this class for integers and Complex class objects. Add any function required in Complex class to support functions of Set class.

**Note:** Create stream insertion operator inside the Set class, otherwise syntax is very tricky

**Task 3: (Aggregation of Set class)**

Create a template class **Sets**, having four set class objects. Create **three** Set objects by sending an integer array having some common & other unique elements around 6-8 for each set object. Take fourth set as Universal set. Add every unique element of 3 sets in universal set using operators defined in Set class. Take complement of three sets using operator defined in Set class. Demonstrate Sets class by writing appropriate main function.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* T e m p l a t e ... ... ... ... ... e t a l p m e T \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*