DHARMSINH DESAI UNIVERSITY, NADIAD
FACULTY OF TECHNOLOGY
B.TECH. SEMESTER II [EC/IT/CE]
SUBJECT: (ES201) NAME: PROGRAMMING FOR PROBLEM SOLVING - II

Examination	**: First Sessional**	**Seat No**	**: 31**
Date	**: 23/03/2023**	**Day**	**: Thursday**
Time	**: 4:00 p.m. to 5:15 p.m.**	**Max. Marks**	**: 36**

INSTRUCTIONS:
1. Figures to the right indicate maximum marks for that question.
2. The symbols used carry their usual meanings.
3. Assume suitable data, if required & mention them clearly
4. Draw neat sketches wherever necessary

Q.1 **Do as directed.** **[12]**

CO2 R **(a)** Fill in the blanks [2]

(I) Requesting to make a function _____, helps eliminate the cost of calls to small functions.

(II) By default, all the members of C++ 'struct' are public and hence struct does not support _____ out of the box.

CO2 E **(b)** Complete the segment given below via recommending **two different approaches** to achieve the task about string comparison. Refer Line#3, if statement required condition in the underline placeholder: [2]

```
int main() {                                                    //Line#1
   string s1="ABC",s2="PQRST";                                  //Line#2
   if( _____ )                                                  //Line#3
      cout<<"String s1 is greater than s2.";                    //Line#4
   else                                                         //Line#5
      cout<<"String s1 is NOT greater than s2.";                //Line#6
   return 0;                                                    //Line#7
}                                                               //Line#8
```

CO1 U **(c)** Summarize the purpose of access specifiers (public, private) with respect to data abstraction and encapsulation. [2]

CO1 E **(d)** What will be the **output/error** in the following codes? [2]

(I)
```
#include<iostream>
using namespace std;
namespace {
   int var  = 10;
}
void fun();
int main() {
   int var = 20;
   cout << var << endl;
   fun();  return 0;
}
void fun() {
   cout << var;
}
```

(II)
```
#include<iostream>
using namespace std;
int &fun() {
   static int x = 10;
   return x;
}
int main() {
   fun()  = 30;
   cout << fun();
   return 0;
}
```

CO2 E **(e)** What will be the **output/error/warning** in the following code? [2]

```
#include<iostream>
using namespace std;
int sum(int n1, int n2, int n3, int n4 = 99);;//Prototype#1
int sum(int n1, int n2, int n3 = 98, int n4);;//Prototype#2
int main() {
   int sum(int n1, int n2, int n3 = 100, int n4 = 200);;//Prototype#3
   void fun(); //Prototype#4
   cout << sum(1, 2, 3 ) << endl << sum(1, 2) << endl;
   fun();    return 0;
}
int sum(int n1, int n2, int n3, int n4) { return n1 + n2 + n3 + n4; }
void fun() { cout << sum(1, 2, 3) << endl<< sum(1, 2) << endl;  }
```

CO1 R **(f)** Explain the C++ I/O with get from/extraction (>>) and insertion/put to (<<) operators by suitable labeled drawing. [2]

Q.2 Attempt ***Any TWO*** from the following questions. [12]

CO1 N (a) (I) What will be the **output/error/warning** in the following code? [3]

```
#include <iostream>
using namespace std;
const int &fun(int &num) {
        --num;
        return num--;
}
int main() {
    int i = 10;
    const int &ret_val = fun(i);
    cout<<i<<" "<<ret_val<<endl,    return
    0;
}
```

CO2 C (II) Create two custom manipulators called **pos_num** and **neg_num** for printing the messages "**Positive Number**" and "**Negative Number**" respectively onto the screen. Initialize an integer array with 5 elements containing positive and negative numbers both. Display all the values and their respective message using custom manipulators depending on whether the value is positive or negative. The value zero need not show any label. [3]

CO2 C (b) Create a C++ program that contains two functions having common name **reverse** - one that takes a C-style string as an argument and reverses it and another that takes an array of integers and reverses the order of its elements. Write a driver main function by calling both functions with appropriate arguments and outputting the results. [6]

CO1 A (c) Write a program that creates a **dynamic array of structures** to store the details of n employees (name, employeeID, designation, and salary). The program should then sort the employees in ascending order of their salaries. Display the information of the employees after sorting. [6]

Q.3 Attempt the following: [12]

CO1 U (a) (I) Briefly discuss the evolution/unfolding of procedural oriented programming to object oriented programming. [2]

CO2 A (II) Implement following user defined procedural oriented function: [2]

bool contains(const string& s, const char* p)

Returns true if and only if this c++ string object "s" contains the specified sequence of char values terminated by null, pointed by p, anywhere within. To develop this use C++ string library functions.

CO1 C (b) Using OOP C++ Abstract Data Type (ADT) concept, develop a class to represent the concept of **Time**(hours,minutes,seconds) using a 24 hour format. Provide functionality: [8]

- To set time in 24 hours format. - To display time in 24 hours format.
- To display time in 12 hours format with a.m. or p.m. label, which internally does nesting of member function display time 24 hours format in the parenthesis.
- main() with sample runs to test Time class object tm1.

Note that Ante meridiem (a.m.) means before midday and post meridiem (p.m.) means after midday. Accept that midnight(00:00:00/24:00:00) is 12:00:00 a.m. and midday 12:00:00 is 12:00:00 p.m. Interestingly 12:59:59 is 12:59:59 p.m. and 13:00:00 is 01:00:00 p.m. Lastly 23:59:59 is 11:59:59 p.m.

OR

Q.3 Attempt the following: [12]

CO1 U (a) (I) Explain polymorphism as a concept of OOP. [2]

CO2 A (II) Explain the working of below C++ code: [2]

```
int main(){                                          //Line#1
        string str("abcdef");                        //Line#2
        while(int length=str.size()){                //Line#3
                cout<<str.at(length-1);              //Line#4
                str.erase(length-1);                 //Line#5
        }                                            //Line#6
 return 0;                                           //Line#7
}                                                    //Line#8
```

CO1 C (b) Using OOP C++ Abstract Data Type (ADT) concept, develop a class to represent the concept of **Length**(units of total inches only). Note that 12 inches = 1 foot. e.g. 70 inches length can be represented as 5'-10" which is 12X5 feet=>60 and 60+10=>70 inches. Provide functionality: [8]

- To store length in units of total inches itself.
- Display length in units of feet and inches accordingly.
- To add external total inches to length's total inches
- To add feet and inches to length's total inches
- To add other length to length's total inches
- main() with sample runs to test Length class with objects len1, len2.