Total No. of Questions – [ ] Total No. of Printed Pages

G.R. No.	

Paper Code- U218-126 (T1)

OCTOBER 2018/ IN-SEM (T1)

S. Y. B. TECH. (COMP.) (SEMESTER - I)

COURSE NAME: Object Oriented Programming COURSE CODE: (CSUA21176)

Solution & Marking Scheme

(PATTERN 2017)

Time: **[1 Hour]** [Max. Marks: **30**]

Q.1)a) OOP Features (Data Abstraction, Encapsulation, Data Hiding,Inheritance, Polymorphism , classes and objects 6 marks

b) Declaration of class and objects 1 mark

Definition of function with Proper logic 3 Marks

Declaration of static variable 2 Marks

```
#include<iostream>
using namespace std;
class vote
{
static int candidates[6];
static int number_of_voters;
public:
void get_number_of_voters();
void count_votes(int i);
void get_result();
};

  void vote:: count_votes(int i)
  {
    candidates[i]++;
    number_of_voters++;
  }
  void vote::get_number_of_voters()
  {
    cout<<"The total number of voters is "<<number_of_voters<<endl;
  }
  void vote:: get_result()
  {
    for(int i=1;i<=5;i++)
    {
```

```
            cout<<i<<" got "<<candidates[i]<<" votes"<<endl;
        }
    }
int vote::candidates[6];
int vote::number_of_voters;
int main()
{
    char ch='y';
    int ballot;
    do
    {
        cout<<"Enter candidate number you want to vote(1 to 5)"<<endl;
        vote obj;
        cin>>ballot;
        if(ballot<1 || ballot>5)
        {
            cout<<"Please enter between 1 to 5"<<endl;
            continue;
        }
        obj.count_votes(ballot);
        cout<<"Do you want to continue voting? (y/n) "<<endl;
        cin>>ch;
     }
    while(ch=='y' || ch=='Y');
    cout<<"The number of votes for each candidate are as follows"<<endl;
    vote result;
    result.get_result();
    result.get_number_of_voters();
}
```

c) 2 differences 4 marks

OR

Q.2) a) Definition 1 mark
Purpose of copy constructor 2 marks
Example (with Syntax) 3 Marks

b) Declaration of class and objects 2 marks
Definition of function with Proper logic 4 Marks

```
#include<iostream>
using namespace std;
class Electric
{
      double unit;
      static int bill;
   public:
Electric();
void accept();
void print_bill();
};
Electric::Electric()
{
cout<<"Default Values of Units are"<<endl;
cout<<"Units 1-100"<<"\t"<<"Rs."<<"\t"<<3<<endl;
cout<<"Units 101-300"<<"\t"<<"Rs."<<"\t"<<7<<endl;
cout<<"Units 301-500"<<"\t"<<"Rs."<<"\t"<<9<<endl;
cout<<"Units 501-1000"<<"\t"<<"Rs."<<"\t"<<11<<endl;
cout<<"Beyond 1000"<<"\t"<<"Rs."<<"\t"<<13<<endl;
}

void Electric::accept()
      {

            cout<<"\n No. Of Units Consumed: ";
            cin>>unit;
      }

void Electric::print_bill()
{      if(unit>=1 && unit<=100)
            bill=unit*3;
      else if(unit>101 && unit<=300)
            bill=unit*7;
else if(unit>301 && unit<=500)
            bill=unit*9;
            else if(unit>501 && unit<=1000)
            bill=unit*11;

else if(unit>1000)
            bill=unit*13;
   cout<<"\n Electricity Bill = "<<bill;
}
int main()
{
```

```
    Electric e;
    int i,cnt;
        e.accept();
        e.print_bill();
    return 0;
}
```

C) 2 differences 4marks

Q3 a) Declaration of class and objects 2 marks
Function for copying and concatenation of two string 2 marks each

```
#include <iostream>
using namespace std;

class String{
        char word[20];
        char bword[40];

 public:
        void GetData();
        void operator =(char[]);
        void operator +(char[]);
void display();
};

void String::GetData(){
 cout<<"Enter the string to evaluate : ";
 cin>>word;
 for(int i=0;i<20;i++){
        //if(word[i]!='\0')
                bword[i]=word[i];
 }
}
void String::display(){
 cout<<"\nThe original string is : "<<word<<"\n\n";
}


void String::operator =(char copy[20]){
 int i=0;
 while(word[i]!='\0'){
        copy[i]=word[i];
        i++;
 }
```

```
 copy[i]='\0';
 cout<<"\nThe copied word is : "<<copy<<"\n\n";
}

void String::operator +(char copy[20]){
 int j,flag=0,i;
 for(i=0;i<40;i++){
        if(bword[i]=='\0'){
                for(j=0;j<20;j++){
                        if(copy[j]!='\0'){
                                bword[i]=copy[j];
                                i++;
                        }
                        else{
                                flag=-1;
                                break;
                        }
                }
        if(flag==-1)
                break;
        }
 }
 bword[i]='\0';
 cout<<"The concatenated word is : "<<bword<<"\n\n";

}

int main(){
 String obj;
 obj.GetData();
 int choice,dummy=0,count=0;
 char copy[20];
 do{
 cout<<"\nWhat would you like to do??\n";
 cout<<"\n1.Copy word\n2.Concatenate\n3.Display\n4.Exit";

 cin>>choice;
 switch(choice){
        case 1:

                {
                        obj=copy;
                        break;
                }
        case 2:
```

```
                {
                        cout<<"\nEnter new string to concatenate : ";
                        cin>>copy;
                        obj+copy;
                        break;
                }
        case 3:
                {
                        obj.display();
                        break;
                }
        case 4:
                        return 0;
        default:
                cout<<"\nPlease enter a valid choice.\n\n";
 }
 }while(1);
 return 0;
}
```

b) public and private derivation 2 marks each

c) Function overloading ii) friend function 2 marks each

Q4) a) Declaration of class and objects 2 marks

Order of execution 4 marks for Program

Base and derived class have their own constructor

Constructor is used to construct the object and destructor is called destroy the object

Order of Execution:

First base constructor is called

Next derived constructor is called

Next destructor of derived class is called

Next destructor of base class is called

```
class base
{
```

```
public:
base()
{
cout<<"Base class constructor is called";
}
~base()
{
cout<<"Base class destructor is called";
}};
class child:public base
{
public:
child()
{
cout<<"child class constructor is called";
}
~child()
{
cout<<"child class destructor is called";
}};
int main()
{
child d;
return 0;
}
```

If base class constructor is not taking argument, then there is no need to define constructor in derived class. If base class is having a constructor with argument then we need to define constructor in derived class and pass argument to base class constructor.

B) Definition	1 mark
Pitfalls of operator overloading (at least 3)	3 marks
C) Concept of re usability	2 marks
Justification with example	2 marksx