Name: Enrolment No:	UPES

UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

End Semester Examination, May 2021

Programme Name: B.Tech –ECE **Semester : VI**
Course Name : Programming Technique **Time : 3 Hr**
Course Code : ECEG 3035 **Max. Marks : 100**

Instructions:

1. **Attempt all the questions (Theory, Numerical, Case study etc.)**
2. **Attempt all questions serially as per Question paper.**
3. **Answer should be neat and clean. Draw a free hand sketch for circuits/tables/schematics wherever required.**
4. **Scan the required answer script and check the resolution carefully before uploading. No other mode of submission is acceptable.**
5. **You are expected to be honest about each attempt which you make to progress in life**

SECTION A [5x6]

S. No.		Marks	CO
Q1			
I	**Which of the following operators can be utilized for Scope Resolution?**	2	CO1
(A)	= =		
(B)	: :		
(C)	->		
(D)	( )		
II	Data hiding is the implementation of abstraction in OOPs. **(True/False).**	1	

III	What is the syntax of the Array of Objects?	**2**	

Q2	**Consider the following code segment.**
	
	// Assume that the sizeof(int) = 4 and sizeof(double) = 8.
	
	class parent
	{
	static int statInt;
	double arr[10];
	void display() { }
	};
	
	class base1: public parent { };
	class base2: public parent { };
	class child: protected base1, protected base2 { };
	
	What will be the size (in Bits) of the object child?
		5	**CO2**

Q3	**Implement the flow of execution of try/catch blocks and predict the output of the following code?**
	
	#include <iostream>
	using namespace std;
	
	int main()
	{
	int x = -1;
	
	// Some code
	cout << "Before try \n";
	try {
	cout << "Inside try \n";
	if (x < 0)
	{
	throw x;
	cout << "After throw (Never executed) \n";
	}
	}
	catch (int x) {
	cout << "Exception Caught \n";
	}
	
	cout << "After catch (Will be executed) \n";
	return 0;
	}
		5	**CO3**

Q4
I
	Fill in the blank:
	
	#include<iostream>
	using namespace std;
	class Test
	{ static int x;
	public:
	void get() { x = 15; }
	void print() { x = x + 20; cout << "x =" << x << endl; }
	};
	_____; // Define static variable 'x'
	
	int main()
	{
	Test o1, o2;
	o1.get();
	o2.get();
	o1.print();
	o2.print();
	return 0;
	}
		3+2	**CO2**

II **(A)** **(B)** **(C)** **(D)**	**In which memory a String is stored, when we create a string using new operator?** Stack String memory Heap memory Random storage space		

Q5	**Fetch the output of the following program using "inline" fuction:**
	
	#include <iostream>
	using namespace std;
	class operation
	{
	int a,b,add,sub,mul;
	float div;
	public:
	void get();
	void sum();
		5	**CO1**

```
        void difference();
        void product();
        void division();
};
inline void operation :: get()
{
        cout << "Enter first value:";
        cin >> a;
        cout << "Enter second value:";
        cin >> b;
}

inline void operation :: sum()
{
        add = a+b;
        cout << "Addition of two numbers: " << a+b << "\n";
}

inline void operation :: difference()
{
        sub = a-b;
        cout << "Difference of two numbers: " << a-b << "\n";
}

inline void operation :: product()
{
        mul = a*b;
        cout << "Product of two numbers: " << a*b << "\n";
}

inline void operation ::division()
{
        div=a/b;
        cout<<"Division of two numbers: "<<a/b<<"\n" ;
}

int main()
{
        cout << "Program using inline function\n";
        operation s;
        s.get();
        s.sum();
        s.difference();
        s.product();
        s.division();
        return 0;
}
```

Q6	**What is the output of the following code?**
	
	#include <iostream>
	using namespace std ;
	namespace Ex { int x = 10; }
	namespace Ex { int y = 10; }
	int x = 5;
	int main()
	{ using namespace Ex ;
	x = y = 50;
	cout << x << " " << y;
	return 0;
	}
	
	(A) **10 10**
	(B) **50 50**
	(C) **5 50**
	(D) **Compilation error: ambiguous reference to variable 'x'**
		5	**CO3**

SECTION B [10 x 5]			
Q7	Write a program in C++ to depict the behavior of dominating a data member and over-riding a member function using the concept of "**class**". Also, indicate the adequate comment against each expression.	**10**	**CO2**
Q8 **(a)** **(b)**	**What are the two ways to avoid ambiguity while handling Multiple Inheritance in C++? Also, illustrate the same using a well executable code.** **Elucidate the following:** i. Member access operator ii. return 1; iii. copy constructor	**5+5**	**CO3**

	iv. friend v. access specifier		

Q9
 	
(a)
 	
(b)
	Why is the size of an Empty class not zero?
	
	There is an interesting rule that says that an empty base class need not be represented by a separate byte. So compilers are free to make optimization in case of empty base classes. As an exercise, try the following program and guess the output:
	
	#include <iostream>
	using namespace std;
	
	class Empty {
	};
	
	class Derived1 : public Empty {
	};
	
	class Derived2 : virtual public Empty {
	};
	
	class Derived3 : public Empty {
	char c;
	};
	
	class Derived4 : virtual public Empty {
	char c;
	};
	
	class Dummy {
	char c;
	};
	
	int main()
	{
	cout << "sizeof(Empty) " << sizeof(Empty) << endl;
	cout << "sizeof(Derived1) " << sizeof(Derived1) << endl;
	cout << "sizeof(Derived2) " << sizeof(Derived2) << endl;
	cout << "sizeof(Derived3) " << sizeof(Derived3) << endl;
	cout << "sizeof(Derived4) " << sizeof(Derived4) << endl;
	cout << "sizeof(Dummy) " << sizeof(Dummy) << endl;
	
	return 0;
	}
		[3+7]	**CO3**

Q10 **(a)** **(b)**	**Enlist the functions declarations that cannot be overloaded in C++.** **Following are the things which a derived class inherits from its parent:** 1) Every data member defined in the parent class (although such members may not always be accessible in the derived class!) 2) Every ordinary member functions of the parent class (although such members may not always be accessible in the derived class!) 3) The same initial data layout as the base class **Implement the aforementioned details via a C++ code**	**4+6**	**CO2**
Q11	**Brief about:** **i. ? :** ii. Dynamic Array iii. Encapsulation iv. Destructor v. Pure virtual function vi. Constructor Initialization list vii. Function Overloading viii. Inheritance ix. Exception Handling x. Run time Polymorphism	**10**	**CO3**
SECTION 'C'[20 MARKS]			
Q12 **(a)** **(b)**	Illustrate the following operators in C++: a) Arithmetic Operators: b) Unary Operators: c) Binary Operators: d) Relational Operators: e) Logical Operators: f) Bitwise Operators: g) Assignment Operators: Understated is the Demonstration of **constant object**. Suggest the output after carefully interpreting the code.	**7+ 5+4+4**	**CO2**

```
#include<iostream>
using namespace std;
class Demo
{
    int value;
    public:
    Demo(int v = 0) {value = v;}
    void showMessage()
    {
        cout<<"Hello World We are Tushar, "
        "Ramswarup, Nilesh and Subhash Inside"
        " showMessage() Function"<<endl;
    }
    void display()const
    {
        cout<<"Hello world I'm Rancho "
        "Baba Inside display() Function"<<endl;
    }
};
int main()
{
   //Constant object are initialised at the time of declaration
using constructor
    const Demo d1;
    //d1.showMessage();Error occurred if uncomment.
    d1.display();
    return(0);
}
```

(c)

This program is all about the implementation of Pre/Post Decrementer. Fill the blank by keeping this in mind so that the given test cases will satisfy:

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

class DClass

 { int data;
   public:
          _____{ } // Define Constructor

         DClass& operator- - ()
          { - -data;
          return _____;
           }

         _____ {
          DClass t(data);
          - -data;
          return _____;    }

         void disp()
            { cout << " " << data ; }
  };
```

	int main()
	{
	int x;
	cin >> x;
	
	DClass obj1(x);
	obj1.disp();
	
	DClass obj2 = obj1- -;
	obj2.disp();
	
	obj2 = - -obj1;
	obj2.disp();
	
	return 0;
	}		

(d)	**Compare the characteristics of Constructor and Destructor in C++.**		