12/5/25 (M)

This question paper contains **20** printed pages]

Roll No. | | | | | | | | | | | |

S. No. of Question Paper : **5775**

Unique Paper Code : **2342011201**

Name of the Paper : **Object-Oriented Programming with C++**

(DSC04)

Name of the Course : **B.Sc. (H) Computer Science**

Semester : **II**

Duration : **3 Hours** Maximum Marks : **90**

(Write your Roll No. on the top immediately on receipt of this question paper.)

Paper has two sections. *All* the questions in Section A are compulsory.

Answer any *four* questions from Section B.

Parts of question must be answered together.

Part A

1. (*a*) Draw flowchart for finding even number out of two numbers. 2

(*b*) Identify the valid/invalid identifiers and write the reason for invalid identifiers : 3

(*i*) `Num_total`

(*ii*) `2ndValue`

(*iii*) `my@variable`

P.T.O.

(*iv*) `firstName`

(*v*) `float`

(*vi*) `myVariable123`

(*c*) **Identify and correct the errors in the following code segments :** 10

(*i*)

```
class Example {
    static int count;
public:
    Example()
    {
        count++;
    }
    void display()
    {
        cout << "Count: " << count << endl;
    }
};
int main()
{
    Example obj1, obj2;
    obj1.display();
    obj2.display();
    return 0;
}
```

(*ii*)
```
class Person {
private:
    string name;
public:
    Person(string n)
    {
        name = n;
    }
    void display()
    {
        cout << "Name: " << name << endl;
    }
};
int main()
{
    Person p("John");
    cout << p.name << endl;
    p.display();
    return 0;
}
```

(*iii*)
```
int main()
{
    string str = "Hello";
    str[5] = 'W';
    cout << str << endl;
    return 0;
}
```

(*iv*)
```
int main ()
{
    int a = 10;
    int b = 0;
    try
    {
        if (b == 0)
            throw "Division by zero error!";
        cout << a / b << endl;
    }
    catch (int e)
    {
        cout << "Exception caught: " << e << endl;
    }
    return 0;
}
```

(v)
```
int main()
{
    int arr[5];
    for (int i = 0; i < =5; i++)
    {
        arr[i] = i * 2;
    }
    for (int i = 0; i < 5; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```

(d) **Give the output for the following code segments :** **10**

(i)
```
class Rectangle
{
    int length;
    int width;
public:
    Rectangle()
    {
        length = 0;
        width = 0;
    }
```

```
Rectangle(int  l,  int  w)
   {
      length  =  l;
      width  =  w;
   }
Rectangle(const  Rectangle  &r)
   {
      length  =  r.length;
      width  =  r.width;
   }
void  display()
   {
cout<<"Length:"<<length<<",Width:"<<width<<endl;
   }
};
int  main()
{
   Rectangle  rect1(10,  5);
   Rectangle  rect2  =  rect1;
   rect2.display();
   return  0;
}
```

(*ii*)
```
#include <iostream>
using namespace std;
class Vehicle
{
public:
   void start()
   {
      cout << "Vehicle started!" << endl;
   }
};
class Car : public Vehicle
{
public:
   void drive()
   {
      cout << "Car is driving!" << endl;
   }
};
```

```
int  main( )
{
   Car  myCar;
   myCar.start( );
   myCar.drive( );
   return  0;
}
```

(*iii*)

```
#include  <iostream>
using  namespace  std;
int  main( )
{
   int  a  =  5;
   int  b  =  2;
   float  result1  =  a  /  b;
   float  result2  =  (float)a  /  b;
   float  result3  =  a  /  (float)b;
   float  result4  =  (float)(a  /  b);
   cout  <<  "result1:  "  <<  result1  <<  endl;
   cout  <<  "result2:  "  <<  result2  <<  endl;
   cout  <<  "result3:  "  <<  result3  <<  endl;
   cout  <<  "result4:  "  <<  result4  <<  endl;
   return  0;
}
```

(*iv*)
```
int main()
{
    int matrix[2][3] = {{1, 2, 3}, {4, 5, 6}};
    int sum = 0;
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 3; j++)
        {
            sum += matrix[i][j];
        }
    }
    cout<<"Sum of all elements: "<<sum << endl;
    cout<<"Element at [1] [2] :"<<matrix[1] [2]<< endl;
    return 0;
}
```

(*v*)
```
int main()
{
    int x = 10;
    int* ptr = &x;
    *ptr = *ptr + 5;
    cout << "Value of x: " << x << endl;
    cout << "Value at ptr: " << *ptr << endl;
    return 0;
}
```

(e) **Write a class** `Student` **with data members** `name, roll and marks`. **Include a constructor and a function** `display` **to display student details.** 5

Part B

2. (a) **Rewrite the following code for counting vowels and alphabets using a switch-case :** 5

```
char *ch = "hello world";
int countv = 0, countalp = 0;
int i = 0;
while (ch [i])
{
  if (ch[i]=='a'||ch[i]=='e'||ch[i]=='i'
    ||ch[i]=='o'||ch[i]=='u')
    countv++;
  else
    countalp++;
  i++;
}
cout << countv << endl;
cout << countalp;
```

(*b*) Write a program to compute the series : 10

$$s = 1+\frac{x}{2}+\frac{x^2}{3}+\frac{x^3}{4}+\ldots\ldots\ldots \text{ upto } n \text{ terms}$$

using command line arguments.

3. (*a*) Write declarations/code for the following : 5

(*i*) A function prototype for function `fsum` having int array, double and char as argument and return type as void.

(*ii*) A function prototype for function `f x` with arguments pointer to double, and 2 dimensional array with size 8 × 10, returning integer value.

(*iii*) Array of floats : {3.4, 5.6, 7.8, 9.1}

(*iv*) Pointer variable for float *x*, *y*

(*v*) Ternary operator for max of two values.

(*b*) Write a program using a function `findmaxmin` with pointer arguments to find the maximum and minimum values from two integers entered by the user. Define the function : 10

```
void findMaxMin(int* a,int* b,int& max,int& min)
```

The function should :

(*i*) Accept two integers via pointers.

(*ii*) Store the maximum in the third argument.

(*iii*) Store the minimum in the fourth argument.

And write `main( )` to call this function for finding minimum and maximum out of two integers `x` and `y`.

P.T.O.

4. (*a*) **Explain the output of the following code segment :** 5

```
class  Shape
{
public:
    void  draw( )
    {
       cout  <<  "Drawing  Shape"  <<  endl;
    }
};
class  Circle:  public  Shape
    {
public:
    void  draw( )
    {
    cout  <<  "Drawing  Circle"  <<  endl;
    }
};
int  main()
    {
       Shape*  s;
       Circle  c;
       s  =  &c;
       s->draw( );
       return  0;
    }
```

(*b*) **Write a program that demonstrates single inheritance as per the description given :** **10**

Define a base class `Person` **with the following :**

- **Data members :** `name (string), age (int)`
- **Member function :** `void getDetails( )` **to input name and age**
- **Member function :** `void showDetails( )` **to display name and age**

Define a derived class `Student` **that inherits from** `Person`**. It should have :**

- **Additional data members :** `rollNo (int), marks (float)`
- **Member function :** `void getStudentData( )` **to input roll number and marks**
- **Member function:** `void showStudentData( )` **to display roll number and marks along with inherited data**

Create an object of class `Student` **in** `main( )` **and call appropriate functions to input and display all details.**

5. (*a*) **Identify and correct the errors in the following code segments :** 5

```
class  Base
{
public:
    int  Var1;
protected:
    int  Var2;
private:
    int  Var3;
public:
    Base()
    {Var1=1;
    Var2=2;
    Var3=3; }
void  display( )
{
cout<<"Display  Base  Variable1:"<<Var1<<  endl;
cout<<"Display  Base  Variable2:"<<Var2<<  endl;
cout<<'Display  Base  Variable3: "<<Var3<<  endl;
}
};
```

```
class  Derived:  public  Base
{
public:
void  display()
{
cout<<"Display  Derived  Variable1:"<<Var1<<  endl;
cout<<"Display  Derived  Variable2:"<<Var2<<  endl;
cout<<"Display  Derived  Variable3:"<<Var3<<  endl;
    }
};
int  main()
{
   Base  baseObj;
   Derived  derivedObj;
   cout<<"Accessing  Base  Class  Object:"<<endl;
   baseObj.display();
   cout<<"Accessing  Derived  Class  Object:"<<endl;
   derivedObj.display();
   return  0;
}
```

(*b*) **Differentiate between the following using proper examples :** 10

(*i*) **Constructor and Destructor**

(*ii*) **While loop and do-while loop**

(*iii*) **Function Overloading and Function Overriding**

(*iv*) **Call by Value and Call by Reference**

6. (*a*) **Give output and explain the functioning of the following code segment :** 5

```
void test(int x) {
    if (x == 0)
      throw 0;
    else if (x < 0)
      throw 'N';
    else if (x > 100)
      throw "Too large!";
    else
      cout << "Valid number: " << x << endl;
}
int main()
```

```
{
    try {
       test  (25);
       test(0);
       test  (150);
       test  (-5);
       }
    catch  (...)
    {
    cout  <<  "Some  exception  caught!"  <<  endl;
    }
return  0;
}
```

(*b*) **Write a program that defines a function** `isPalindrome (string str)` **to check whether a given string is a palindrome or not. A string is a palindrome if it reads the same forwards and backwards.** 5

The program should :

- **Take a string input from the user.**
- **Pass it to the function.**
- **Print whether it is a palindrome or not.**

(c) **What are streams ? What will be the content of the file** `data.txt` **and explain the reason after execution of this program ?** 5

```
#include  <fstream>
using  namespace  std;
int  main()
{
    ofstream  fout1("data.txt");
    fout  <<  "Line  1\n";
    fout  <<  "Line  2";
    fout.close( );
    ofstream  fout2("data.txt");
    fout2  <<  "New  Line";
    fout2.close();
    return 0;
}
```

7. (a) **What is pure virtual function ? Give output of the following code segment :** 5

```
#include  <iostream>
using  namespace  std;
class  Account
{
public:
   virtual  void  balance() =  0;
};
```

```
class SavingsAccount : public Account
{

   void balance ()
   {
   cout<<'Savings Account Balance: $1000"<< endl;
   }
};
class CurrentAccount: public Account
{
public:
   void balance()
     {
     cout<<"Current Account Balance: $500" << endl;
     }
};
int main()
{
   Account* acc;
   SavingsAccount savings;
   CurrentAccount current;
   acc = &savings;
   acc->balance();
   acc = &current;
   acc->balance();
   return 0;
}
```

(*b*) Give output and explain the functioning of the following code segment : 5

```
template <typename T>
T add(T a, T b) {
    return a + b;
}
int main() {
    cout << add(5, 10) << endl;
    cout << add(3.5, 2.1) << endl;
    return 0;
}
```

(*c*) Write a program to check whether the numeber is prime or not. 5