**Final Term Examination-**Fall-2017

**Object Oriented Programming (CS-423/CS-727)**

**BS (CS/IT), MCS, MIT**

**Total Time: 1.5 Hrs . Maximum Marks: 30 .**

**Registration No.\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**--------------------------------------------------------------------------------**

**Question 01**: Complete the following classes with minimum required variables and functions. **[6]**

class Shape { //do not delete any thing

protected: float area; float perimeter;

public: virtual void display();

virtual string getShapeName() = 0; //This function will return the name of shape

};

class Rectangle: public Shape {

public:

Rectangle(float width, float height);//This Constructor will calculate & save ***area*** & ***perimeter***

};

class Circle: public Shape {

public: Circle(float radius); //This Constructor will calculate & save ***area*** and ***perimeter*** of circle

};

**Question 02**: Assuming there is no error in main() complete the class ***UIIT\_Calc*** and its functions. **[8]**

int main()

{ UIIT\_Calc c1;

try {

c1.getData(); //get ***Number1*** and ***Number2*** in character array[40] and ***operator*** in char

c1.Validate1(); //if Number1 has characters 0-9 or dot(.) then assign it to class member double

// ***Operand1*** else throw exception which shows message "Error:" << ***BadOpd*** <<

//" is not a valid number". The builtin ***isdigit(Number1[i])*** function will help you

//to check its number or not & ***atof(Number1)*** will help you convert into double.

c1.Validate2();

try {

c1.Calculator(); //This function will use switch case to display the result for four

//operators **+**, **-** , **\*** and **/** . For all other operators it throws exception and

} //displays “Operation Error: " << ***e2.op*** << " is not a valid operator".

catch( UIIT\_Calc::DivError e1)

{

cout << "\nBad Operation: " << e1.str; //Will display “Bad Operation: ***Div by 0 not allowed***”

}

}

catch( UIIT\_Calc::OpError e2)

{ cout << "\nOperation Error: " << e2.op << " is not a valid operator";

}

catch( char \*BadOpd)

{ cout << "\nError: " << BadOpd << " is not a valid number";

}

return 0;

}

**Question 03**: On Back Side of Question Paper

**Question 04**: On Back Side of Question Paper

**---P.T.O.---**

**Question 03**: Assuming there is no error in main() complete the class ***Rectangle*** and its functions. **[8]**

int main ()

{

Rectangle rect1(3.4,5.68); //both values will save in ***length*** and ***width*** variables of class

Rectangle rect2(2.4,4.68);

if(rect1--!=++rect2) cout<<"\n Both are not of same size";

else cout<<"\n Both are of same size";

rect1.show();//will display values 2.4,4.68

Rectangle rect3=(rect1, rect2);

rect3.show();//will display values 3.4,5.68

return 0;

}

**Question 04**: Assuming there is no error in main() complete the class ***MyClass*** and all functions. **[8]**

int main ()

{

MyClass <double, float, double> c1(3,2.0,3.0);

//class has 3 variables, ***no\_of\_items***, ***length*** and ***width***. no\_of\_items always int

MyClass <float, double, int> c2(2,10.5,2);

MyClass <double, float, double> c3(1,3.0,2.0);

cout<<c1.calc\_area(); //will display double value 18.0

cout<<c2.calc\_area(); //will display float value 42.0

c3=add\_area(c1,c3); //will add all 3 respective variables of c1 and c3

cout<<c3.calc\_area(); //will display double value 100.0

return 0;

}

**\*\*\*\*\*Good Luck\*\*\*\*\***

**Final Term Examination-**Fall-2017

**Object Oriented Programming (CS-423/CS-727)**

**PRACTICAL**

**BS (CS/IT), MCS, MIT**

**Total Time: 1.0 Hr . Maximum Marks: 20**

**Registration No.\_\_\_\_\_\_\_\_\_\_\_\_\_**

**--------------------------------------------------------------------------------**

**Question 01**: **[14]**

**Complete the following C++ class of Timetable Management system by adding the following functions:**

**class Timetable**

**{**

**int T\_ID;**

**string course\_title;**

**string start\_time;**

**float class\_duration;**

**string teacher\_name;**

**int room\_no;**

**public:**

**- - - - -**

**- - - - -**

**};**

1. login() //take password from user and verifies with the stored in file “**password.txt**”(3)
2. getRecord( ) // **5** variables from user and T\_ID is initialized with autoincremented value (2)
3. insertRecord( ) // insert record as an object in append mode in “**TimeRecord.txt**” (3)
4. retrieveRecord( ) // retrieve all record from file “**TimeRecord.txt**”. (3)
5. searchRecord( ) // search and display the records based on ***teacher\_name*** from file. (3)

**Question 02**: **[6]**

**Discuss your project of Object Oriented Programming course in 5-6 lines.**

**\*\*\*\*\*Good Luck\*\*\*\*\***