

## B.E in INFORMATION TECHNOLOGY

2<sup>ND</sup> YEAR, 1<sup>ST</sup> SEMESTER SUPPLEMENTARY EXAMINATION, 2023

## OBJECT ORIENTED PROGRAMMING

Time: 3 hours

Full Marks: 100

<p>CO1</p> <p>[15 MARKS]</p>	<p>1. a) Which of the following function prototypes are valid? State reasons. For each of the valid prototypes, show how many ways the function can be invoked.</p> <pre>int sum(int i=1, int j, int k); int sum(int i=1, int j=2, int k); int sum(int i=1, int j, int k=3); int sum(int i, int j=2, int k=3); int sum(int i, int j=2, int k); int sum(int i, int j, int k=3);</pre> <p>b) State 2 circumstances where <b>this</b> operator is used. Illustrate with suitable code snippets.</p> <p>c) Distinguish between normal function and inline function. State some scenarios where a function is not executed as <b>inline</b> even if it is declared as inline.</p> <p>d) Can a function call statement appear on the left hand side of an assignment operator? If no, state why. If yes, show how.</p> <p style="text-align: right;">[6+3+(2+1)+3=15]</p>
<p>CO2</p> <p>[30 MARKS]</p>	<p>2. a) Consider the following code snippet having the definition of a class <b>Number</b>. Keeping the class definition unchanged, define all the methods and constructor outside the class definition and show how they are invoked from <b>main()</b> method. Discuss the outputs.</p> <p>Point out the drawback of the code and state how this drawback can be overcome.</p> <pre>class Number { int a; public: Number( int=0); void add(Number); friend void swap(Number, Number); void show(); };</pre> <p>b) Justify the truth or falsity of each of the following statements. Provide code snippets, where/if necessary, for justification.</p> <ol style="list-style-type: none"> <li>A class can have several constructors and destructors.</li> <li>A copy constructor of a class takes the photocopy of object of that class.</li> <li>A non-static member method of a class can access only the non-static member variables of that class.</li> <li>Method overloading is also known as early binding.</li> <li>A constant member function of a class can access only the constant member variables of a class.</li> </ol> <p>c) Which property of OOP is not maintained by friend functions and why?</p> <p>Write a complete C++ program to define a class <b>Time</b> with 2 data members: <b>hour</b> and <b>minute</b> and create two objects from that class T1 and T2. Now define a friend function of <b>Time</b> class to perform addition between the objects T1 and T2 and print the result as follows.</p> <p><b>Example:</b> 3 hr 40 min (T1) + 2 hr 30 min (T2)= 6 hr 10 min (result)</p> <p style="text-align: right;">[(5+1+2)+(3x5)+(2+5)=30]</p>

[ Turn over

<p><b>CO3</b> <b>[20 MARKS]</b></p>	<p>3. a) Suppose a class <i>Employee</i> has a private member variable <i>basic_sal</i> and a member function <i>get_sal()</i>. Two subclasses <i>Manager</i> and <i>Clerk</i> have been derived from it. These two types of employees receive 40% and 30% allowances respectively on their basic monthly salary. Show with a suitable code how to compute the total salary of these two types of employees. What type of inheritance does it refer?</p> <p>b) Assume a class <i>Base</i> with a parameterized constructor and a destructor. Create a derived class <i>Child</i> inheriting the <i>Base</i> class having its own constructor and destructor. Show (with code snippet) how the constructor of the <i>Base</i> class is invoked through <i>Child</i> class. State in what order these constructors and destructors of the two classes are invoked. Discuss in this connection, the drawback of such destructor and how this drawback can be overcome.</p> <p>c) Suppose two classes <i>Base1</i> and <i>Base2</i> have a public member function <i>display()</i> each within them. A <i>Child</i> class has been inherited from these two classes. Show (with proper code) how the <i>Base2</i> class version of the <i>display()</i> function is invoked through the <i>Child</i> class object. What type of inheritance does it refer?</p> <p>d) Distinguish between virtual function and pure virtual function with code snippets.</p> <p style="text-align: right;">[(5+1)+(4+2+2)+(3+1)+2=20]</p>
<p><b>CO4</b> <b>[20 MARKS]</b></p>	<p>4. a) Consider the following class <i>Sample</i>. Modify the class with suitable code to carry out the operations as mentioned in the <i>main()</i> method. Indicate the value of the variable <i>t</i> which will be printed.</p> <pre> class Sample {int a;  public: //add suitable member methods };  int main() {Sample s1(5), s2, s3(10); s2=s1(++s3); s2=s3+3; int t=s2; cout&lt;&lt;"t="&lt;&lt;t; //invoke display() method to print the values of the member variables of s1, s2 and s3 } </pre> <p>b) Consider the following class <i>Sample</i>. Update its definition to perform the tasks as specified in <i>main()</i>. Clearly indicate which portion of the class is dedicated for which task. Finally discuss the output.</p> <pre> class Sample { int s; };  int main() { Sample ob1(3), ob2(4); ob1+=ob2; ob1++.show(); ob2=10+ob1; cout&lt;&lt;ob1&lt;&lt;ob2; } </pre> <p>c) Consider a namespace NS having the declaration of a function <i>display()</i> and the definition of a class <i>Sample</i>. This <i>Sample</i> class is also having the declaration of a member function with the same signature as that of the <i>display()</i> method. Now show how these two versions of the <i>display()</i> method can be defined outside the namespace definition.</p> <p style="text-align: right;">[8+8+4=20]</p>
<p><b>CO5</b> <b>[15 MARKS]</b></p>	<p>5. a) Illustrate how exceptions of integer, float, character and string types are thrown from one <i>try</i> block and caught in different <i>catch</i> blocks. Show how to replace these multiple <i>catch</i> blocks by a single one.</p> <p>b) When does the need of template specialization arise? Show with suitable code snippet.</p> <p>c) Show (with a snippet) how an exception can be re-thrown?</p> <p>d) How can you supply default parameter/s in template functions? Show with suitable code snippet.</p> <p style="text-align: right;">[(4+2)+3+3+3=15]</p>