



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

Important Instructions to examiners:

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No	Sub Q.N.	Answer	Marking Scheme
1.	(A) (a) Ans.	Attempt any SIX of the following: What is scope resolution operation? Scope resolution operation includes scope resolution operator . It is used to uncover a hidden variable. Scope resolution operator allows access to the global version of a variable. The scope resolution operator is used to refer variable of class anywhere in program. :: Variable_name OR Scope resolution operator is also used in classes to identify the class to which a member function belongs. Scope resolution variable is used to define function outside of class. return_type class_name :: function_name() { } Define pointer. Give syntax for declaration of pointer.	12 2M <i>Correct explanation 2M</i>
	(b) Ans.		2M



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		Definition: Pointer is a variable that holds memory address of another variable of similar data type. Syntax to declare pointer variable: data_type *pointer_variable;	<i>Correct definition 1M</i> <i>Correct syntax 1M</i>
	(e) Ans.	What is copy constructor? Copy constructor is a constructor which creates an object by initializing it with an object of the same class, which has been created previously. The copy constructor is used to: <ul style="list-style-type: none">• Initialize one object from another of the same type.• Copy an object to pass it as an argument to a function.• Copy an object to return it from a function.	2M <i>Correct explanation 2M</i>
	(d) Ans.	Define polymorphism. Enlist its types. Definition: Polymorphism means ability to take more than one form that means a program can have more than one function with same name but different behavior. Types of polymorphism: 1) Compile time polymorphism 2) Runtime polymorphism	2M <i>Correct definition 1M</i> <i>Correct two types 1M</i>
	(e) Ans.	List various visibility modes used in inheritance. Different visibility modes used in inheritance are: 1) Private 2) Protected 3) Public	2M <i>Any two correct visibility modes 1M each</i>
	(f) Ans.	What are objects? How are they created? Objects Objects are basic run time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program has to handle. An object is the instance of the class . Objects are created in following way: class classname { class definition	2M <i>Correct explanation of object 1M</i>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>}object;</pre> <p style="text-align: center;">OR</p> <pre>class classname { class definition }; main() { classname object; }</pre>	<p><i>Correct description of creation of object</i> 1M</p>
	<p>(g) Ans.</p>	<p>Write use of 'This' pointer. this pointer is used to represent an object that invokes a member function. It points to the object for which the function is called. It is also used to access members of object inside function definition of called function.</p>	<p>2M <i>Correct use</i> 2M</p>
	<p>(h) Ans.</p>	<p>What do you mean by default argument? Give its suitable example. Default argument Initializing an argument with a value while defining a constructor is referred as constructor with default value. When a constructor with default value is declared in a class, it does not require object to pass value for default argument. Constructor will execute without passing default argument value with the object. If object contains value for default argument, then passed value overwrites the default value.</p> <p>Example: class ABC { int x,y; public: ABC(int p, int q=10) { x=p; y=q; } }; void main() {</p>	<p>2M <i>Correct explanation of default argument</i> 1M</p> <p><i>Any suitable example</i> 1M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>ABC obj1(5); ABC obj2(20,30); }</pre>	
1.	(B) (a) Ans.	<p>Attempt any TWO of the following:</p> <p>Explain multiple constructors in a class with suitable example.</p> <p>Multiple constructors in a class means a class can contain more than one constructor. This is also known as constructor overloading. All constructors are defined with the same name as the class they belong to. All the constructors contain different number of arguments. Depending upon the number of arguments, the compiler executes appropriate constructor.</p> <p>Multiple constructors can be declared in different ways:</p> <pre style="margin-left: 20px;">integer(); // Default Constructor(No arguments) integer(int, int); // Parameterized Constructor(Two arguments)</pre> <p>When an object is created the first constructor is invoked.</p> <p>In the first case, the constructor itself supplies the data values and no values are passed by the calling program.</p> <p>In the second case, the function call passes the appropriate values from main() to the constructor.</p> <p>Example:</p> <pre>#include<iostream.h> #include<conio.h> class integer { int m, n; public: integer() { m = 0; n = 0; } // constructor 1 default constructor integer(int a, int b) { m = a; n = b; cout<<"value of m="<<a; cout<<"value of n="<<b; } // constructor 2 parameterized constructor };</pre>	8 4M <i>Correct explanation 2M</i> <i>Suitable example 2M</i>

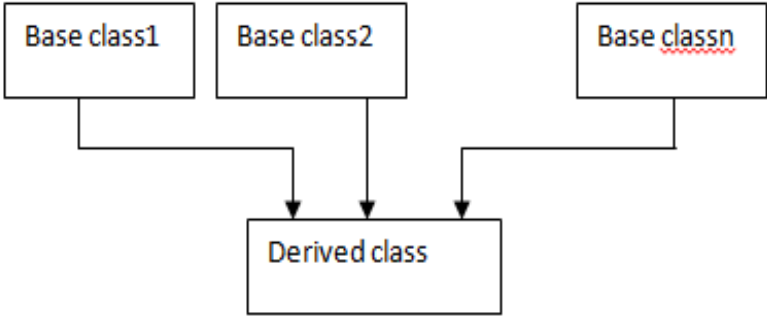
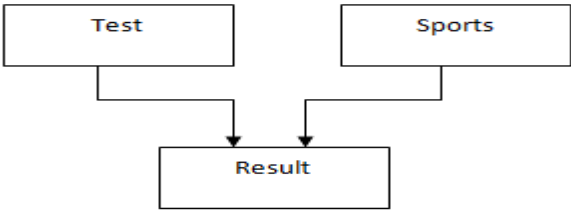


MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
 (Autonomous)
 (ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>void main() { clrscr(); integer i1; //calls constructor 1 integer i2(20,40); // calls constructor2 getch(); }</pre> <p>In the above example, constructor is overloaded by defining two constructors in the same class. Both the definitions are different with respect to number of arguments. The first constructor does not accept any argument and the second constructor accepts two integer arguments.</p>	
(b) Ans.	<p>Define multiple inheritance. Give example.</p> <p>Multiple Inheritance: When a single class is derived from more than one base class then it is known as multiple inheritance. A derived class can inherit the attributes of all base classes from which it is derived.</p> <p><i>Syntax:</i></p>  <pre> graph TD BC1[Base class1] --> DC[Derived class] BC2[Base class2] --> DC BCn[Base classn] --> DC </pre> <p><i>Example:</i></p>  <pre> graph TD T[Test] --> R[Result] S[Sports] --> R </pre>		<p style="text-align: center;">4M</p> <p style="text-align: center;"><i>Correct Definition 2M</i></p> <p style="text-align: center;"><i>Any correct example 2M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>class Test { }; class Sports { }; class Result:publicTest,public Sports { };</pre> <p>In the above example class 'Result' is a single derived class derived from two base classes base class 'Test' and base class 'Sports'.</p>	
(c) Ans.	<p>Explain destructor with suitable example.</p> <p>Destructor:</p> <ol style="list-style-type: none">1. A destructor is a special member function whose task is to destroy the objects that have been created by constructor.2. It does not construct the values for the data members of the class.3. It is invoked implicitly by the compiler upon exit of a program/block/function.4. Destructors are not classified in any types.5. Destructor never accepts any parameter.6. Destructor name is preceded with tilde operator. <p>Syntax:</p> <pre>~classname() {.... }</pre> <p>Example:</p> <pre>#include<iostream.h> #include<conio.h> class time { private: int hrs, mins,sec; public: time(int h,int m,int s) { hrs=h; mins=m;</pre>	<p>4M</p> <p><i>Correct explanation 2M</i></p> <p><i>Relevant example 2M</i></p>	



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>sec=s; } ~time() // Destructor { cout<<"hours deleted"; cout<<"minutes deleted"; cout<<"seconds deleted"; } void display() { cout<<"hours="<<hrs; cout<<"Minutes="<<mins; cout<<"seconds="<<sec; } }; void main() { time t(2,43,56); t.display(); getch(); }</pre>	
2.	(a) Ans.	<p>Attempt any FOUR of the following:</p> <p>How to define a member function outside the body of class?</p> <p>The user can declare member function outside the class with the help of scope resolution operator (::). The label class_name:: tells the compiler that the function_name belongs to the class class_name.</p> <p>Syntax:</p> <pre>return_type class_name :: function_name(argument(s)) { Function body; }</pre> <p>Example:</p> <pre>void student::accept() { cout<<"Enter the student's data:"; cin>>roll_no>>name; }</pre>	16 4M <i>Correct explanation 4M</i>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		In above example accept() member function of class student is defined outside of the class.	
(b) Ans.	Explain the concept of virtual function with example. Virtual Function: A virtual function is a member function that is declared within a base class and redefined by its derived class. When base class and its derived class both contain same name member function then derived class function overrides base class function. Base class pointer is used to refer member functions of its class as well as its derived class. When base pointer is used to refer to functions, it ignores the contents of the pointer and selects the member function that matches the function call. To execute derived class version of the overridden function virtual keyword is used with base class function. When a function is made virtual, compiler checks the address stored inside the pointer. If the pointer points to base class then function from base class is executed. If it contains address of derived class then function from derived class is executed. Run time polymorphism requires virtual function to execute same name function from base class and derived class depending on address stored inside the pointer. Example: #include<iostream.h> class Base { public: virtual void show() { cout<<"\n show base"; } }; class Derived : public Base { public: void show() { cout<<"\n show derived"; } }; void main()	4M <i>Correct explanation 2M</i> <i>Any correct example 2M</i>	



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>{ Base B,*bptr; Derived D; bptr=&B; bptr->show(); bptr=&D; bptr->show(); }</pre> <p>In above example, both base and derived class contains same name function as show. By creating a pointer of base class one can invoke desired show function by storing address of respective object in pointer.</p>	
(c) Ans.	<p>What is the purpose of 'protected' access specifier used in C++?</p> <p>Protected access specifier:</p> <ol style="list-style-type: none">1. Protected access specifier is mainly used in inheritance in C++.2. It uses protected keyword defined in C++ language.3. Class members declared as protected can be accessed by the member functions within its class and any class immediately derived from it.4. These members cannot be accessed by the functions outside these two classes. <p>Example:</p> <pre>class base { protected: int b; public: void display() { cout<<b; } }; class derived:public base { public: void show() { cout<<b;</pre>	4M <i>Correct explanation 4M</i>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>} }; void main() { derived d; d.display(); d.show(); }</pre> <p>In above example variable 'b' can be accessed by its member function 'display ()' as well as member function 'show()' of its derived class as it is a protected member.</p>	
<p>(d) Ans.</p>	<p>Give advantages of object oriented approach over procedure oriented approach.</p> <p>Advantages of object oriented approach over procedure oriented approach:</p> <ol style="list-style-type: none">1. In object oriented approach, through inheritance we can eliminate redundant code and extend the use of existing classes.2. We can build programs from the standard working modules that communicate with one another, rather than having to start writing the code from scratch. This leads to saving of development time and higher productivity.3. The principle of data hiding helps the programmer to build secure programs that cannot be invaded by code in other parts of the program.4. It is possible to have multiple instances of an object to coexist without any interference.5. It is possible to map objects in the problem domain to those in the program.6. It is easy to partition the work in a project based on objects.7. The data centered design approach enables us to capture more details of a model in implementable form.8. Object oriented systems can be easily upgraded from small to large systems.9. Message passing techniques for communication between objects makes the interface descriptions with external systems much simpler.10. In OOP software complexity can be easily managed.	<p>4M</p> <p><i>Any 4 correct points 1M each</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>(e) Ans.</p>	<p>Explain friend function. Give example. Friend function: The private members of a class cannot be accessed from outside the class but in some situations two classes may need access of each other's private data. So a common function can be declared which can be made friend of more than one class to access the private data of more than one class. The common function is made friendly with all those classes whose private data need to be shared in that function. This common function is called as friend function. Friend function is not in the scope of the class in which it is declared. It is called without any object. The class members are accessed with the object name and do membership operator inside the friend function. It accepts objects as arguments.</p> <p>Example: #include <iostream.h> #include <conio.h> class abc { int a; public: void get1() { cin>>a; } friend void add(abc,xyz); }; class xyz { int a; public: void get1() { cin>>a; } friend void add(abc,xyz); }; void add(abc a1,xyz x1) {</p>	<p>4M</p> <p><i>Correct explanation 2M</i></p> <p><i>Any relevant example 2M</i></p>
--	---------------------	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>cout<<a1.a+x1.a; } void main() { abc a1; xyz x1; a1.get1(); x1.get1(); add(a1,x1); }</pre>	
	(f) Ans.	<p>Explain the concept of pointer to derived classes.</p> <p>Pointer to derived class:</p> <p>Pointers can be used to point to the base class objects and objects of derived class. Pointers to objects of base class are type-compatible with pointers to objects of a derived class. Single pointer variable can be made to point objects belonging to different classes.</p> <p>If B is base class and D is derived class then pointer declared as a pointer to B can also be a pointer to D.</p> <p>Example:</p> <pre>B *cptr; // pointer of base class B b;//Base object D d;// Derived object cptr=&b; //cptr stores address of object b of base class cptr=&d; //cptr stores address of object d of derived class</pre> <p>Using cptr, pointer of base class type, we can access only those members which are inherited from B and not the members that originally belong to D. In case a member of D has the same name as one of the members of B, then any reference to that member by cptr will always access the base class member.</p>	4M <i>Correct explanation 4M</i>
3.	(a) Ans.	<p>Attempt any FOUR of the following:</p> <p>What is dynamic memory allocation? Explain with example.</p> <p>Allocating memory at run time (when program is in execution) is called as dynamic memory allocation.</p> <p>Dynamic memory allocation use malloc () and calloc () functions to allocate memory at run time. C++ supports these two functions as well as it defines unary operator new to perform the task of allocating memory.</p>	16 4M <i>Relevant Explanation 2M</i>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>An object of any type can be created by using new operator. To declare an object with new following syntax is used: Pointer_variable=new data_type;</p> <p>The new operator allocates sufficient memory to hold a data object of type data_type and returns the address of an object. The return address is stored inside pointer_variable.</p> <p>Example:</p> <pre>class sample { private: int a; int b; public: void getdata() { cin>>a>>b; } void putdata() { cout<<a<<b; } }; main() { sample *ptr=new sample; ptr->getdata(); ptr->putdata(); }</pre>	<p><i>Any correct example</i> 2M</p>
<p>(b) Ans.</p>	<p>Explain parameterized constructors with example. A constructor that can take arguments is known as parameterized constructor. In some applications, it may be necessary to initialize the various data members of different objects with different values when they are created. So parameterized constructor is used to achieve this by passing arguments to the constructor function when the objects are created.</p>	<p>4M <i>Explanation</i> 2M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>Example:</p> <pre>class ABC { int m; public: ABC(int x) //parameterized constructor { m=x; } void put() { cout<<m; } }; void main() { ABC obj(10); // call to parameterized constructor obj.put(); }</pre> <p>In the above example, constructor ‘ABC (int x)’ is a parameterized constructor function that accepts one parameter. When ‘obj’ object is created for class ‘ABC’, parameterized constructor will be invoked and data member ‘m’ will initialize with the value 10 which is passed as an argument. Member function ‘put’ displays the value of data member ‘m’.</p>	<p><i>Any correct example</i> 2M</p>
<p>(c) Ans.</p>	<p>Explain virtual base class with suitable example.</p> <p>Consider a situation where all three kinds of inheritance, namely, multilevel, multiple, hierarchical inheritance, are involved. This illustrated in fig a. the child has two direct base classes, “parent1” & “parent2” which themselves have a common base class “grandparent”. The child inherits the traits of “grandparent” via two separate path . It can also inherit directly as shown by broken line. The “grandparent” sometimes referred to as indirect base class.</p>	<p>4M</p> <p><i>Explanation of virtual base class</i> 2M</p>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p><i>Example:</i></p> <div style="text-align: center;"> <pre> classDiagram class Grandparent class Parent1 class Parent2 class Child Grandparent -- > Parent1 Grandparent -- > Parent2 Parent1 -- > Child Parent2 -- > Child Grandparent .. > Child </pre> <p style="text-align: center;">Fig. a: Virtual Base Class</p> <p>Inheritance by the “child” as shown in fig a might pose some problems. All the public & protected members of “grandparent” is inherited into “child” twice, first via “parent1” & again via “parent 2”. This means, “child” would have duplicate sets of the members inherited from “grandparent. This introduces ambiguity & should be avoided. The duplication of inherited members in child class due to these multiple paths can be avoided by making the common base class as virtual base class while declaring the direct or intermediate base classes as shown above.</p> <pre> class Grandparent { }; class Parent1: public virtual Grandparent { }; class Parent2: public virtual Grandparent { }; class Child: public Parent1,public Parent2 { }; </pre> </div>	<p><i>Any correct example 2M</i></p>
<p>(d) Ans.</p>	<p>Write a program to overload ‘+’ operator to concatenate two strings.</p> <pre> #include<iostream.h> #include<string.h> #include<conio.h> class string1 { char str[20]; </pre>	<p>4M</p> <p><i>Correct logic 2M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>public: void getdata() { cout<<"\n Enter String :"; cin>>str; } void display() { cout<<str; } void operator+(string1 x) //Concatenating String { strcat(str,x.str); } }; void main() { string1 str1, str2; clrscr(); str1.getdata(); str2.getdata(); str1+str2; cout<<"\n\n Concatenated String is:"; str1.display(); getch(); }</pre>	<p><i>Syntax</i> 2M</p>
<p>(e) Ans.</p>	<p>Explain pointer to object in detail. When address of an object of a class is stored into the pointer variable of the same class type then it is pointer to object. This pointer can be used to access the data member and member functions of same class. When a pointer is used, the arrow operator (->) rather than the dot operator is employed. To declare an object specify its class name, and then precede the variable name with an asterisk. Syntax: - class_name *pointer_variable; To obtain the address of an object, precede the object with the & operator. Syntax: - pointer_variable=&object_name;</p>	<p>4M</p> <p><i>Relevant explanation</i> 4M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>Example: #include<conio.h> #include<iostream.h> class product { private: int code; public: void getdata(void) { cout<<"Enter code:"; cin>>code; } void display(void) { cout<<"\nCode="<<code; } }; void main() { clrscr(); product p1; product *ptr; ptr=&p1; ptr->getdata (); ptr->display(); getch(); }</pre>	
<p>(f) Ans.</p>	<p>Explain class with suitable example. A class is a user defined data type which binds data and its associated functions together. It allows the data and functions to be hidden, if necessary from external use. Generally, a class specification has two parts. i) Class declaration: it describes the type & scope of its members. ii) Class function definitions: It describes how the class functions are implemented.</p> <p>Class declaration.</p>	<p>4M</p> <p><i>Explanation</i> 2M</p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>The general form of a class is</p> <pre>class class-name { private: variable declaration; function declaration; public: variable declaration; function declaration; };</pre> <p>1) The class keyword specifies that what follows is an abstract data of type class name. The body of a class is enclosed within braces & terminated by semicolon.</p> <p>2) The class body consists of declaration of variables & functions which are called as members & they are grouped under two sections i.e. private & public.</p> <p>3) Private and public are known as visibility labels, where private can be accessed only from within the class where public members can be accessed from outside the class also. By default, members of a class are private.</p> <p>4) The variable declared inside the class are known as data members & functions are known as member functions. Only the member function can have access to the private data members & private functions. However the public members can be accessed from outside the class.</p> <p>Example:</p> <pre>class item { int number; float cost; public : void getdata (int a, float b); void putdata (void); };</pre> <p>In above example, class-name is item. These class data members are private by default while both the functions are public by declaration. The function getdata() can be used to assign values to the member variable number & cost, and putdata() for displaying their values.</p>	<p><i>Any correct example 2M</i></p>
--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		These functions provide the only access to data members of the class.	
4.	(a)	<p>Attempt any FOUR of the following: Identify the inheritance shown in fig. 1 implement it by using suitable member function.</p> <div style="text-align: center;"> <pre> classDiagram class Student { Roll No. Name } class Exam { Subject Name } class Library { Member - No. } Student < -- Exam Student < -- Library </pre> <p>(Fig.-1)</p> </div> <p>Ans. Inheritance given in fig1 is hierarchical inheritance.</p> <pre> #include <iostream.h> #include<conio.h> class Student { protected: int Roll_No; char Name[20]; public: void accept() { cout<< "\nEnter Roll no and name:\n"; cin>>Roll_No>> Name; } void display() { cout<<"Name ="<< Name<<" "<<"Roll no="<<Roll_No; } }; class Exam : public Student { protected: char Subject_Name[20]; public: void getdata() </pre>	<p align="right">16 4M</p> <p align="right"><i>Identify 1M</i></p> <p align="right"><i>Correct impleme ntation with member function 3M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

```
{
    cout<< "\nEnter Subject Name= ";
    cin>>Subject_Name;
}
void display_sname()
{
    cout<<"Subject name="<<Subject_Name;
}
};
class Library : public Student
{
    protected:
        int Member_No;
    public:
        void getno()
        {
            cout<< "\nMember_No= ";
            cin>>Member_No;
        }
        void display_no()
        {
            cout<<"Member number="<<Member_No;
        }
};
int main()
{
    clrscr();
    Exam e; //object of derived class B
    Library lib; //object of derived class C
    e.accept();
    e.display();
    e.getdata();
    e.display_sname();

    lib.accept();
    lib.display();
    lib.getno();
    lib.display_no();
    getch();
}
```



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		}	
(b)	Ans.	<p>Write any four characteristics of constructor.</p> <p>The constructor functions have some special characteristics.</p> <ul style="list-style-type: none"> They should be declared in the public section. They are invoked automatically when the objects are created. They do not have return types, not even void and therefore, and they cannot return values. They cannot be inherited, though a derived class can call the base class constructor. Like other C++ functions, they can have default arguments. Constructors cannot be virtual. We cannot refer to their addresses. An object with a constructor (or destructor) cannot be used as a member of a union. They make 'implicit calls' to the operators new and delete when memory allocation is required. 	<p>4M</p> <p><i>Any four characteristics</i> 1M each</p>
(c)	Ans.	<p>Explain data types in C++. <i>(Note: Diagram is optional)</i></p> <div style="text-align: center;"> <pre> graph TD Root[C++ Data Types] --> UserDefined[User-defined type] Root --> BuiltIn[Built-in type] Root --> Derived[Derived type] UserDefined --> UserDefinedList[structure union class enumeration] BuiltIn --> Integral[Integral type] BuiltIn --> Void[Void] Integral --> Int[int] Integral --> Char[char] Derived --> Floating[Floating type] Derived --> Array[array] Derived --> Function[function pointer] Derived --> Reference[reference] Floating --> Float[float] Floating --> Double[double] </pre> </div> <p>Primitive Built-in Types: C++ offer the programmer a rich assortment of built-in as well as user defined data types. Following table lists down seven basic C++ data types: Following table lists down seven basic C++ data types:</p>	<p>4M</p> <p><i>Explanation</i> <i>(Any four data types)</i> 4M</p>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

Type	Keyword
Boolean	bool
Character	char
Integer	Int
Floating point	float
Double floating point	double
Valueless	void
Wide character	wchar_t

Integer: Keyword used for integer data types is int. Integers typically requires 4 bytes of memory space and ranges from -2147483648 to 2147483647.

Character: Character data type is used for storing characters. Keyword used for character data type is char. Characters typically requires 1 byte of memory space and ranges from -128 to 127 or 0 to 255.

Boolean: Boolean data type is used for storing boolean or logical values. A boolean variable can store either true or false. Keyword used for boolean data type is bool.

Floating Point: Floating Point data type is used for storing single precision floating point values or decimal values. Keyword used for floating point data type is float. Float variables typically requires 4 byte of memory space.

Double Floating Point: Double Floating Point data type is used for storing double precision floating point values or decimal values. Keyword used for double floating point data type is double. Double variables typically require 8 byte of memory space.

void: Void means without any value. void datatype represents a valueless entity. Void data type is used for that function which does not returns a value.

Wide Character: Wide character data type is also a character data type but this data type has size greater than the normal 8-bit datatype. Represented by wchar_t. It is generally 2 or 4 bytes long.

User defined data types:

Structure: It is a collection of related data elements that belong to



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<p>similar or different data type.</p> <p>Class: It is a collection of data members and member functions that operates on data.</p> <p>enum: An enumerated data type provides a way for attaching names to numbers. The enum keyword (form C) automatically enumerates a list of words by assigning them values 0, 1, 2, and so.on.</p> <p>Derived Data Type:</p> <p>Array: It is a collection of similar data type elements.</p> <p>Function: It is a collection of statements written to execute a specific task.</p> <p>Pointer: It is a variable that stores address of another variable of similar data type.</p>	
	<p>(d) Ans.</p>	<p>Explain static member function.</p> <p>Static member function: - a static member function can have access to only other static variables or functions declared in the same class. It can be called using the class name instead of its object. It can be declared inside the class with static keyword placed before return type.</p> <p>Syntax for declaration:-</p> <pre>static return_type function_name () { function body }</pre> <p>Syntax for calling static function:-</p> <pre>class_name::function_name();</pre> <p>Example:</p> <pre>class test { static int count;-----static data member public: void setcount() -----member function { count=count+1; } static void showcount()-----static member function {</pre>	<p style="text-align: center;">4M</p> <p style="text-align: center;"><i>Relevant explanation 4M</i></p>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>cout<<count; } }; int test::count; void main() { test t1,t2; t1.setcount(); t2.setcount(); test::showcount(); -----Call to static member function. }</pre>	
(e) Ans.	<p>Explain single inheritance with suitable example. When a single derived class is derived from only one base class then it is called as single inheritance.</p> <div style="text-align: center;"> <pre> graph TD BC[Base Class] --> DC[Derive Class] </pre> </div> <p>In a single inheritance, derived class can inherit some or all members of base class. It is implemented by specifying base class name and visibility mode proceeded with colon symbol while declaring derived class as shown below.</p> <pre>class base { }; class derived: public base { };</pre> <p>Example: <pre>#include<iostream.h> #include<conio.h> class College</pre></p>		<p style="text-align: center;">4M</p> <p style="text-align: center;"><i>Explanation 2M</i></p> <p style="text-align: center;"><i>Any correct example</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>{ protected: int a; }; class Student: public College { public: void getdata() { cin>>a; } void putdata() { cout<<a; } }; void main() { Student s; clrscr(); s.getdata(); s.putdata(); getch(); }</pre>	2M										
(f) Ans.	<p>Explain searching elements in array using pointers. Consider an array of five elements as shown below: A[5]={ 10,20,30,40,50};</p> <table border="1"><thead><tr><th>A[0]</th><th>A[1]</th><th>A[2]</th><th>A[3]</th><th>A[4]</th></tr></thead><tbody><tr><td>10</td><td>20</td><td>30</td><td>40</td><td>50</td></tr></tbody></table> <p>Search element(SE)=30 Pointer variable is declare as *ptr; Before starting search process initialize pointer variable with the address of first element in an array. ptr=&a[0]; Compare *ptr(value at address stored in ptr) with search element in every iteration. After each iteration ptr points to next location in an array.</p>	A[0]	A[1]	A[2]	A[3]	A[4]	10	20	30	40	50	4M <i>Relevant explanat ion 4M</i>
A[0]	A[1]	A[2]	A[3]	A[4]								
10	20	30	40	50								



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<p>Iteration1: Check *ptr==SE</p> <table border="1" style="margin-left: auto; margin-right: auto; text-align: center;"> <tr><td>A[0]</td><td>A[1]</td><td>A[2]</td><td>A[3]</td><td>A[4]</td></tr> <tr><td>10</td><td>20</td><td>30</td><td>40</td><td>50</td></tr> </table> <p style="text-align: center;">↑ *ptr!=SE Ptr=ptr+1 //As both are not equal increment pointer by 1</p> <p>Iteration1: Check *ptr==SE</p> <table border="1" style="margin-left: auto; margin-right: auto; text-align: center;"> <tr><td>A[0]</td><td>A[1]</td><td>A[2]</td><td>A[3]</td><td>A[4]</td></tr> <tr><td>10</td><td>20</td><td>30</td><td>40</td><td>50</td></tr> </table> <p style="text-align: center;">↑ *ptr!=SE Ptr=ptr+1 //As both are not equal increment pointer by 1</p> <p>Iteration1: Check *ptr==SE</p> <table border="1" style="margin-left: auto; margin-right: auto; text-align: center;"> <tr><td>A[0]</td><td>A[1]</td><td>A[2]</td><td>A[3]</td><td>A[4]</td></tr> <tr><td>10</td><td>20</td><td>30</td><td>40</td><td>50</td></tr> </table> <p style="text-align: center;">↑ *ptr=SE</p> <p>Stop search process as both are equal.</p> <p>If search element is not present in an array, then after comparing all elements stop the search process.</p>	A[0]	A[1]	A[2]	A[3]	A[4]	10	20	30	40	50	A[0]	A[1]	A[2]	A[3]	A[4]	10	20	30	40	50	A[0]	A[1]	A[2]	A[3]	A[4]	10	20	30	40	50	
A[0]	A[1]	A[2]	A[3]	A[4]																													
10	20	30	40	50																													
A[0]	A[1]	A[2]	A[3]	A[4]																													
10	20	30	40	50																													
A[0]	A[1]	A[2]	A[3]	A[4]																													
10	20	30	40	50																													
5.	<p>(a) Ans.</p>	<p>Attempt any FOUR of the following:</p> <p>How to achieve compile time polymorphism explain in detail. The process of linking of function call to function definition at compile time is called as Compile Time Polymorphism. It can be through Function and operator overloading.</p> <p>Function overloading: The process of defining the function with same name but with different number or type of argument is known as Function Overloading. In function Overloading, the function would perform different operations depending on the argument list in the function call. The correct function to invoked is determined at compile time by checking the number and type of the arguments but not on the function return type.</p>	<p>16 4M</p> <p style="text-align: right;"><i>Explanation of any one method</i> 4M</p>																														



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p><i>Example:</i> Function Overloading: Swapping of two integers and swapping of two float values. // Function definition1- swapping two integers void swap(int*p, int*q) { int t; t=*p; *p=*q; *q=t; } // Function definition2- swapping two floats void swap(float*p,float*q) { float t; t=*p; *p=*q; *q=t; } For the function call swap (&a, &b) where a and b are integers and function definition1 will be executed and function definition2 for the function call as swap (&a, &b) when a and b are floats.</p> <p style="text-align: center;">OR</p> <p>Operator Overloading: The Process of defining operator function to extend the use of existing operator to operate on User-defined data type such as ‘object of class’ is known as Operator Overloading.</p> <p><i>Example:</i> Overloading + operator to concatenate two strings. class string1 { char str[20]; public: void getdata() { cout<<"\n Enter String :"; cin>>str; }</p>	
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre> void display() { cout<<str; } void operator+(string1 x) //Concatenating String { strcat(str,x.str); } }; void main() { string1 str1, str2; clrscr(); str1.getdata(); str2.getdata(); str1+str2; cout<<"\n\n Concatenated String is:"; str1.display(); getch(); } </pre>																						
(b) Ans.	<p>Compare structure and class.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Sr. No.</th> <th style="width: 45%;">Structure</th> <th style="width: 45%;">Class</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>Structure contains logically related data items which can be of similar type or different type.</td> <td>Class is a way of binding data and functions together in one single unit. It is a collection of data members and member functions.</td> </tr> <tr> <td style="text-align: center;">2</td> <td>Members of a structure are public by default.</td> <td>Members of a class are private by default</td> </tr> <tr> <td style="text-align: center;">3</td> <td>In structure data is not hidden from external use.</td> <td>It allows data and functions to be hidden from external use.</td> </tr> <tr> <td style="text-align: center;">4</td> <td>It does not support inheritance.</td> <td>It supports inheritance.</td> </tr> <tr> <td style="text-align: center;">5</td> <td>In Structures, structure variable is created that contains data</td> <td>In class object is created that contains data and has access to member functions.</td> </tr> <tr> <td style="text-align: center;">6</td> <td>Declaration: struct structure_name</td> <td>Declaration: class class_name</td> </tr> </tbody> </table>		Sr. No.	Structure	Class	1	Structure contains logically related data items which can be of similar type or different type.	Class is a way of binding data and functions together in one single unit. It is a collection of data members and member functions.	2	Members of a structure are public by default.	Members of a class are private by default	3	In structure data is not hidden from external use.	It allows data and functions to be hidden from external use.	4	It does not support inheritance.	It supports inheritance.	5	In Structures, structure variable is created that contains data	In class object is created that contains data and has access to member functions.	6	Declaration: struct structure_name	Declaration: class class_name	<p>4M</p> <p style="text-align: center;"><i>Any four points 1M each</i></p>
Sr. No.	Structure	Class																						
1	Structure contains logically related data items which can be of similar type or different type.	Class is a way of binding data and functions together in one single unit. It is a collection of data members and member functions.																						
2	Members of a structure are public by default.	Members of a class are private by default																						
3	In structure data is not hidden from external use.	It allows data and functions to be hidden from external use.																						
4	It does not support inheritance.	It supports inheritance.																						
5	In Structures, structure variable is created that contains data	In class object is created that contains data and has access to member functions.																						
6	Declaration: struct structure_name	Declaration: class class_name																						



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<pre>{ type struct_element 1; type struct_element 2; ... type struct_element N; };</pre>	<pre>{ data member; member function; };</pre>	
	<p>(c) Ans.</p>	<p>Write a program to demonstrate the use of pure virtual function. Consider the following example where parent class provides an interface to the base class to implement a function called getArea() as pure virtual function:</p> <pre>#include <iostream.h> // Base class class Shape { protected: int width; int height; public: virtual intgetArea() = 0; void setWidth(int w) { width = w; } void setHeight(int h) { height = h; } }; // Derived classes class Rectangle: public Shape { public: intgetArea() { return (width * height);</pre>		<p>4M</p> <p><i>Correct program 4M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>} }; class Triangle: public Shape { public: intgetArea() { return (width * height)/2; } }; void main(void) { Rectangle Rect; Triangle Tri; Rect.setWidth(5); Rect.setHeight(7); // Print the area of the object. cout<< "Total Rectangle area: " <<Rect.getArea() <<endl; Tri.setWidth(5); Tri.setHeight(7); // Print the area of the object. cout<< "Total Triangle area: " <<Tri.getArea() <<endl; }</pre>	
(d) Ans.	<p>State the concepts of object oriented programming.</p> <p>Basic Concepts of Object Oriented Programming:</p> <p>1. Objects Objects are the basic run time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program has to handle. An object is the instance of the class. When a program is executed, the objects interact by sending messages to one another.</p> <p>2. Classes A class is the collection of related data and function under a single name. A class is collection of object of similar type. The entire set of data and code of an object can be made a user-defined data type with the help of class. Once a class has been defined, we can create any</p>	4M <i>Any 4 concepts 1M each</i>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>number of objects belonging to that class. Classes are user-defined that types and behave like the built-in types of a programming language.</p> <p>3. Data Abstraction and Encapsulation The wrapping up of data and function into a single unit (called class) is known as encapsulation. The data is not accessible to the outside world, and only those functions which are wrapped in the class can access it. This insulation of the data from direct access by the program is called data hiding or information hiding. Abstraction refers to the act of representing essential features without including the background details or explanation. Classes use the concept of abstraction; they encapsulate all the essential properties of the object that are to be created.</p> <p>4. Inheritance Inheritance is the process by which objects of one class acquired the properties of objects of another classes. In OOP, the concept of inheritance provides the idea of reusability. This means that we can add additional features to an existing class without modifying it. This is possible by deriving a new class from the existing one. The new class will have the combined feature of both the classes.</p> <p>5. Polymorphism Polymorphism means the ability to take more than one form. An operation may exhibit different behavior in different instances. For example, consider the operation of addition. For two numbers, the operation will generate a sum. If the operands are strings, then the operation would produce a third string by concatenation.</p> <p>6. Dynamic Binding Binding refers to the linking of a procedure call to the code to be executed in response to the call.</p> <p>7. Message Passing An object-oriented program consists of a set of objects that communicate with each other. Objects communicate with one another by sending and receiving information.</p>	
(e) Ans.	Explain pointer arithmetic with example. C++ allows pointers to perform the following arithmetic	4M



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<p>operations:</p> <ul style="list-style-type: none">a. A pointer can be incremented (++) or decremented (--)b. Any integer can be added or subtracted from a pointer.c. One pointer can be subtracted from another. <p>For example:</p> <pre>int a[5], *p; p = &a[0] OR p = a;</pre> <p>here, p refers to the base address (address of 1st element) of the array variable a.</p> <p>We can increment pointer variable as follows: p++; or ++ p; This statement moves the pointer to the next memory address. Similarly, we can decrement pointer variable as p --; or -- p; this statement moves the pointer variable to the previous memory location.</p> <p>Increment or decrement pointer variable depends on data type of pointer. if pointer variable is of type int then it increments or decrements by two positions in an array.</p> <p>Program to illustrate pointer arithmetic:</p> <pre>#include<iostream.h> #include<conio.h> void main () { int num[5]={56,75,22,18,90}; int *ptr, i; // Declration of pointer variable ptr ptr=&num; cout<<"array elements are::"; for(i=0;i<5;i++) { cout<<*ptr << "\n"; ptr=ptr+1; } ptr=num; // Initializing the base address to the ptr cout<<"value of ptr:"<<*ptr; // Printing the value in the array cout<<"\n"; ptr++; cout<<"value of ptr++:"<<*ptr;</pre>	<p><i>Relevant explanat ion 2M</i></p> <p><i>Any correct example 2M</i></p>
--	--	---



**SUMMER – 2019 EXAMINATION
 MODEL ANSWER**

Subject: Object Oriented Programming

Subject Code: 17432

		<pre> cout<<"\n"; ptr--; cout<<"value of ptr--:"<<*ptr; cout<<"\n"; ptr=ptr+2; cout<<"value of ptr+2:"<<*ptr; cout<<"\n"; ptr=ptr-1; cout<<"value of ptr-1:"<<*ptr; cout<<"\n"; ptr+=3; cout<<"value of ptr+=3:"<<*ptr; cout<<"\n"; getch(); } </pre> <p>OUTPUT: array elements are: 56 75 22 18 90 value of ptr: 56 value of ptr++: 75 value of ptr--: 56 value of ptr +2: 22 value of ptr-1: 75 value of ptr+3: 90</p>										
(f) Ans.	<p>Differentiate between static binding and dynamic binding.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Sr. No.</th> <th style="width: 40%;">Static binding</th> <th style="width: 40%;">Dynamic binding</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">1</td> <td>The linking of function call to function definition at compile time is known as static binding.</td> <td>The linking of function call to function definition at run time is known as dynamic binding</td> </tr> <tr> <td style="text-align: center;">2</td> <td>Function definition to be</td> <td>Function definition to be</td> </tr> </tbody> </table>		Sr. No.	Static binding	Dynamic binding	1	The linking of function call to function definition at compile time is known as static binding.	The linking of function call to function definition at run time is known as dynamic binding	2	Function definition to be	Function definition to be	4M <i>Any four differen</i>
Sr. No.	Static binding	Dynamic binding										
1	The linking of function call to function definition at compile time is known as static binding.	The linking of function call to function definition at run time is known as dynamic binding										
2	Function definition to be	Function definition to be										



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

			<p>executed is selected/ linked based on number or type of argument passed with function call at compile time.</p>	<p>executed is selected by checking the content of base class pointer at run time.</p>	<i>ces 1M each</i>
		3	Complete function definition is available at the time of function call.	Completer function definition is not available at the time of function call	
		4	It is faster in execution.	It is slower in execution	
		5	Implemented with Function overloading and operator overloading	Implemented with virtual function	
6.	<p>(a)</p> <p>Ans.</p>	<p>Attempt any TWO of the following:</p> <p>Explain object as a function argument using following points with suitable example:</p> <p>(i) Pass by value</p> <p>(ii) Pass by reference</p> <p>(i) Pass by Value:</p> <p>When an object is passed by value to a function, a copy of that object is created and changes are reflected on the copy object not on original object.</p> <p>Example:</p> <pre>#include<iostream.h> #include<conio.h> class Example { int x; public: Example(int a) { x=a; } void print() { cout<<"x="<<x; }</pre>			<p>16</p> <p>8M</p> <p style="text-align: center;"><i>Each function 4M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>void swap(Example e) { int t; t=e.x; e.x=x; x=t; } }; void main() { Example e1(4),e2(5); clrscr(); cout<<"Before swapping\n"; cout<<"value of e1:"; e1.print(); cout<<"value of e2:"; e2.print(); e1.swap(e2); cout<<"\nAfter swapping\n"; cout<<"value of e1:"; e1.print(); cout<<"value of e2:"; e2.print(); getch(); } /*****OUTPUT*****/ Before swapping value of e1:x=4value of e2:x=5 After swapping value of e1:x=5value of e2:x=5</pre> <p>(ii) Pass by reference: If we want the called function work with the original object so that there is no need to create and destroy the copy of it, we may pass the reference of the object. Then the called function refers to the original object using its reference or alias and changes will be directly reflected on original copy of the object.</p>	
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

		<p>The following program illustrates it:</p> <pre>#include<iostream.h> #include<conio.h> class Example { int x; public: Example(int a) { x=a; } void print() { cout<<"x="<<x; } void swap(Example &e) { int t; t=e.x; e.x=x; x=t; } }; void main() { Example e1(4),e2(5); clrscr(); cout<<"Before swapping\n"; cout<<"value of e1:"; e1.print(); cout<<"value of e2:"; e2.print(); e1.swap(e2); cout<<"\nAfter swapping\n"; cout<<"value of e1:"; e1.print(); cout<<"value of e2:"; e2.print();</pre>	
--	--	---	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>getch(); } /*****OUTPUT*****/ Before swapping value of e1:x=4value of e2:x=5 After swapping value of e1:x=5value of e2:x=4</pre>	
<p>(b) Ans.</p>	<p>Explain constructor in derived class with suitable example.</p> <p>When a class is declared, a constructor can be declared inside the class to initialize data members. When a base class contains a constructor with one or more arguments then it is mandatory for the derived class to have a constructor and pass arguments to the base class constructor. When both the derived and base classes contain constructors, the base constructor is executed first and then the constructor in the derived class is executed.</p> <p>The constructor of derived class receives the entire list of values as its arguments and passes them on to the base constructors in the order in which they are declared in the derived class.</p> <p>General form to declare derived class constructor: Derived-constructor (arglist1,arglist(D)):Base1(arglist1)</p> <pre>{ Body of derived class constructor }</pre> <p>Derived constructor declaration contains two parts separated with colon (:). First part provides declaration of arguments that are passed to the derived constructor and second part lists the function calls to the base constructors.</p> <p>Example:</p> <pre>#include<iostream.h> #include<conio.h> class base { int x; public: base(int a) { x=a;</pre>	<p>8M</p> <p><i>Explanation 4M</i></p> <p><i>Any correct example 4M</i></p>



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>cout<<"Constructor in base. x="<<x; } }; class derived: public base { int y; public: derived(int a,int b):base(a) { y=b; cout<<"Constructor in derived.y="<<y; } }; int main() { clrscr(); derived ob(2,3); getch(); return 0; } </pre> <p>In the above example, base class constructor requires one argument and derived class constructor requires one argument. Derived class constructor accepts two values and passes one value to base class constructor.</p>	
(c)	<p>Write a program using concept of pointers to string for performing following operations:</p> <p>(i) String concatenation (ii) String comparisons <i>(Note: Single program with both operation shall be considered)</i></p> <p>Ans.</p> <p>(i) Program to implement String Concatenation:</p> <pre>#include<iostream.h> #include<conio.h> void main() { char s1[50],s2[30],*p,*q; clrscr(); cout<<"\n enter string s1 and s2\n"; cin>>s1>>s2; </pre>	8M
		<i>Correct</i>



SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>p=s1; q=s2; while(*p!=NULL) { p++; } while(*q!=NULL) { *p=*q; p++; q++; } *p='\0'; cout<<"\n Concatenated string:"<<s1; getch(); }</pre> <p>(ii) Program to implement String Comparison:</p> <pre>#include<iostream.h> #include<conio.h> void main() { char s1[50],s2[30],*p,*q; int flag=0,c1=0,c2=0; clrscr(); cout<<"\n Enter string s1 and s2\n"; cin>>s1>>s2; p=s1; q=s2; while(*p!='\0') { c1++; p++; } while(*q!='\0') { c2++; q++; }</pre>	<p><i>program for String Concate nation 4M</i></p> <p><i>Correct program for string compari son 4M</i></p>
--	--	--



MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER – 2019 EXAMINATION
MODEL ANSWER

Subject: Object Oriented Programming

Subject Code: 17432

	<pre>} p=s1; q=s2; if(c1!=c2) { flag=1; } else { while(*p!='\0') { if(*p==*q) { p++; q++; } else { flag=1; break; } } } if(flag==1) cout<<"Strings are not equal"; else cout<<"Strings are equal"; getch(); }</pre>	
--	---	--