Lecture Plan 1

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section - A

S. No.	Topic :-Introduction to C++, Differences between OOP & POP	Time Allotted:-
1.	Introduction	—
	- C++ is an Object Oriented Programming Language (C with classes),was developed by Bjarne Stroustrup at AT&T Bell Laboratories	10
	- C++ has best of 2 Languages, i.e. Simula67 & C.	
	- C++ is superset of C.	
2	Division of the Topic	
	- What is C++?	10
	- Benefits of OOP	10
	- Difference between POP & OOP	10
3.	Conclusion	
	The 3 important facilities that C++ adds on to C are Classes, Function Overloading,& Operator Overloading. These features enables us to create abstract data types,inherit properties from existing data types & support Polymorphism, thus making C++ truly Object Oriented.	5
4	Question / Answer	
	• How does C++ help with the tradeoff of safety vs. usability?	5
	• Why we have started using C++ when we had such a powerful language called C?	

Assignment to be given:-
Basic Features of OOPS

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 2

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - A

S. No.	Topic :- Structure of C++ program, directives, data types & declaration	Time Allotted:-
1.	Introduction - The **preprocessor directive** `#include` to include the contents of the header file `iostream.h` in the program. `Iostream.h` is a standard C++ header file and contains definitions for input and output. - This line defines a **function** called `main`. A function may have zero or more **parameters**; A function may also have a **return type.** - This line is a **statement.** This statement causes the **string** `"Hello world\n"` to be sent to the `cout` **output stream**.	10
2	Division of the Topic - Structure of C++ program - Preprocessor directive - Data types & declaration	10 10 10
3.	Conclusion So, a structure of a C++ program includes directives, header files, data types & declaration.	5
4	Question / Answer • What are the various data types available in c++? • Explain the structure of a c++ program?	5

Assignment to be given:-

1. Basic structure of c++ program
2. What are different types of pre-processor directives?

Reference Readings:-

C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 3

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - A

S. No.	Topic :-Header files & Namespaces, Library files, function	Time Allotted:-
1.	Introduction - A **function definition** consists of two parts: interface and body. The **interface** of a function specifies how it may be used - A namespace allows you to group names in your program together to provide some context for them and to avoid conflicts with names declared in other namespaces. - Header files to define the |data types, classes, functions, macros, and constructs	10
2	Division of the Topic	
	- Header files	5
	- Namespaces	5
	- Library files	5
	- Function	
3.	Conclusion Header files & Namespaces, Library files, function are important part of a C++ program structure	10
4	Question / Answer - Name any five header files with their use. - Why namespaces are used? - Explain the use of delay() function?	15

Assignment to be given:-

1. Difference between header file and Library file.
2. Different methods of parameter passing

Reference Readings:-

C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 4

Faculty:-Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section- A

S. No.	Topic: - Structure Definition, accessing members of structure. Introduction to classes & objects, features of OOPS.	Time Allotted:-
1.	Introduction - A class is a logical abstraction, but an object has physical existence. - A `class` defines a data type, much like a `struct` would be in C.	10
2	Division of the Topic - Structure Definition - Accessing members of structure - Introduction to classes & objects - Features of OOPS	5 5 5 5
3.	Conclusion When using well-designed classes, object-oriented programming can significantly increase code reuse and make your programs easier to maintain and extend. Programming with classes and objects differs from ordinary structured programming in these important ways: - Function and operator overloading - Encapsulation of data and methods**.** - Inheritance**.** - Aggregation.	10
4	Question / Answer • What is a class? • What is an object? • When is an interface "good"? • What is encapsulation? • How can I prevent other programmers from violating encapsulation by seeing the `private` parts of my class?	5

Assignment to be given:-

1. Is Encapsulation a Security device?
2. What's the difference between the keywords `struct` and `class`?

Reference Readings:-

C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 5

Faculty:- Nidhi Jindal   Semester:-III   Class:- ECS   Course Code:- IT-202 F

Subject: - OOPS   Section - A

S. No.	Topic: - Encapsulation, access modifiers, polymorphism, overview of inheritance, abstract classes & class's behavior.	Time Allotted:-
1.	Introduction Access-specifier can be: **`public:`** Allow functions or data to be accessible to other parts of the program. **`private:`** May be accessed only by other members of the class. **`protected:`** Used when inheritance is involved; discussed later.	10
2	Division of the Topic	
	- Encapsulation	5
	- Access modifiers--- public, private, protected, default	5
	- Polymorphism	5
	- Overview of inheritance	5
	- Abstract classes	5
	- Class's behavior	2
3.	Conclusion Inheritance is used for two purposes. • We often find that objects share common behavior. • A second use of inheritance is to allow general purpose operations, such as those for lists, to be applied to all objects which belong to sub-classes of that for which the operation is defined in the list class.	5
4	Question / Answer • Explain all concepts of OOPS? • Explain the various types of inheritance	5

Assignment to be given:-
1. Use of access specifiers.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 6

Faculty:- Nidhi Jindal   Semester:-III   Class:- ECS   Course Code:- IT-202 F

Subject: - OOPS   Section - B

S. No.	Topic: - Class scope & accessing class members, separating interface from implementation.	Time Allotted:-
1.	Introduction There are five scopes: - global scope (resolved by `::`) - class scope (resolved by `classname::`) - namespace scope (resolved by `name::`) - file scope (within a .cpp file) - block scope (within `{` and `}`)	10
2	Division of the Topic - Class scope - Accessing class members - Separating interface from implementation.	5 5 5
3.	Conclusion A class is a description of one type of object. In analysis or design it defines: • the potential states that the object may reach; • the external appearance of the object in terms of operations it supplies and their arguments; • the effects of each of the operations, for each possible set of argument values which that operation can accept, on each state of the object;	10
4	Question / Answer • How to access private & public members of a class? • What is protected? Where do we use it?	15

Assignment to be given:-
1. Different types of scopes with example.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 7

Faculty:-Nidhi Jindal   Semester:-III   Class:- ECs   Course Code:- IT-202 F

Subject: - OOPS   Section - B

S. No.	Topic :- Controlling access functions & utility functions, initializing class objects: constructors.	Time Allotted:-
1.	Introduction - The constructor has the same name exactly as its class. The destructor has the same name exactly, preceded by a tilde. - Construction occurs at the point where its declaration is reached, for an automatic variable of a class type, or when new is used to allocate space in the heap, for a class pointer. - Destruction occurs when the block containing its declaration terminates, for an automatic class variable, or when delete is used to return to the heap the space allocated to a pointer to a class.	10
2	Division of the Topic - Controlling access functions & utility functions - Initializing class objects: constructors	15 15
3.	Conclusion Constructors are used to initialize objects of a class, while destructors perform reverse function by de-initializing the objects of a class.	10
4	Question / Answer • What's the deal with constructors? • Can one constructor of a class call another constructor of the same class to initialize the object? • Why can't I initialize my `static` member data in my constructor's initialization list?	5

Assignment to be given:-
1. What is a constructor? What are its types.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 8

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - B

S. No.	**Topic :-** Using default arguments with constructors. Using destructors.	**Time Allotted:-**
1.	Introduction - C++ requires that all classes have a default constructor, that is to say a constructor that can be used without arguments. Track::Track(): fMass(0.), fMomentum(0.), fEnergy(0.) {} The empty argument list shows this to be the default constructor.	10
2	Division of the Topic - Using default arguments with constructors - Using destructors.	15 15
3.	Conclusion Constructors are of many types. We have: - Default constructors - Constructors with parameters - Copy constructors	10
4	Question / Answer • What's the deal with destructors? • What's the order that local objects are destructed? • What's the order that objects in an array are destructed?	5

Assignment to be given:-

1. Can I overload the destructor for my class?
2. Should I explicitly call a destructor on a local variable?

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 9

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - B

S. No.	**Topic: - Constant object & constant member functions, object as member of classes.**	**Time Allotted:-**
1.	Introduction - A constant member function is a function which can not change the values of parameters which have been passed through it. - While a constant object is an object whose value can't be changed. It is fixed. - A class's object can be declared as a member of another class.	10
2	Division of the Topic - Constant object & constant member functions - Object as member of classes.	 15 10
3.	Conclusion So, constant object and constant member functions are used where we do not want changes to occur.	10
4	Question / Answer • Can we have constant class?	5

Assignment to be given:-

1. Write a Program to show the use of constant classes and constant objects.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 10

Faculty:-Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section - B

S. No.	**Topic: - Friend function & friend classes. Using This pointer. Dynamic memory allocation with new & delete.**	**Time Allotted:-**
1.	Introduction Friend functions are the functions which are used to access private members of a class. Friend function may or may not be a member of class. In order to request dynamic memory, the operator **new** exists. *new* is followed by a data *type* and optionally the number of elements required within brackets `[]`.	10
2	Division of the Topic - Friend function & friend classes - Using This pointer - Dynamic memory allocation with new & delete.	5 5 5
3.	Conclusion Since the necessity of dynamic memory is usually limited to concrete moments within once it is no longer needed it should be freed so that it becomes available for future req dynamic memory.	10
4	Question / Answer • What is a `friend`? • Do friends violate encapsulation? • What are some advantages/disadvantages of using `friend` functions? • What does it mean that "friendship isn't inherited, transitive, or reciprocal"? • Should my class declare a member function or a `friend` function?	15

Assignment to be given:-
1. Write a program to show the use of new, delete and this pointer.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 11

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section - B

S. No.	Topic: - Static class members, container classes & integrators, proxy classes, function overloading.	Time Allotted:-
1.	Introduction Class can contain static members, either data or functions. Static data members of a class are also known as "class variables", because their content does not depend on any object. There is only one unique value for all the objects of that same class. Introduction to container classes & integrators, proxy classes & function overloading.	10
2	Division of the Topic - Static class members - Container classes & integrators - Proxy classes - Function overloading.	5 5 5 5 5
3.	Conclusion Static variables are initialized to zero as soon as they are created. Their scope is throughout the class. Containers, integrators & algorithm are integrated i.e., they are correlated to each other.	10
4	Question / Answer • Why should I use container classes rather than simple arrays? • How can I build a <favorite container> of objects of different types?	5

Assignment to be given:-
1. Write a Program to show the use of static variable and static member function.
Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 12

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section -C

S. No.	**Topic: - Introduction to operator overloading, Restriction's on operator overloading.**	**Time Allotted:-**
1.	Introduction Operating overloading allows you to pass different variable types to the same function different results. Operator overloading is common-place among many efficient C++ Programmers. It allows you to use the same function name, but as different functions.	10
2	Division of the Topic - Introduction to operator overloading - Restriction's on operator overloading	15 5
3.	Conclusion Operator overloading can be a powerful programming tool when it increases usability and understandability of a class.	10
4	Question / Answer • What are the benefits of operator overloading? • What are some examples of operator overloading?	10

Assignment to be given:-
1. List the operators that can be overloaded and the operators that cannot be overloaded

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 13

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - C

S. No.	Topic: - Operator functions as class members vs as friend functions.	Time Allotted:-
1.	Introduction ■ Operator functions ❑ Member functions - Use this keyword to implicitly get argument - Gets left operand for binary operators (like +) - Leftmost object must be of same class as operator ❑ Non member functions (friend function) - Need parameters for both operands - Can have object of different class than operator - Must be a friend to access private or protected data	10
2	Division of the Topic	
	- Operator functions as class members	15
	- Operator functions as friend functions.	5
3.	Conclusion - Operator overloading, when used incorrectly increases the burden on the programmer and the client. - When implemented correctly, programs look “neat” and concise.	10
4	Question / Answer • What operators can/cannot be overloaded? • What are some guidelines / "rules of thumb" for overloading operators?	5

Assignment to be given:-
1. Write a program to read a string and overload the ‘+’ operator to add 2 strings.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 14

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section - C

S. No.	Topic: - Overloading <<, >> operators, overloading unary & binary operators.	Time Allotted:-
1.	Introduction - Binary Operators The compiler will treat an expression of the form: a @ b where a is a class object, @ is the operator, and b is the only explicit argument passed to the function. Therefore, it will be treated like: a.operator@(b) - Unary Operators The compiler will treat an expression of the form: @a	10
2	Division of the Topic	
	- Overloading <<, >> operators	15
	- Overloading unary & binary operators.	15
3.	Conclusion Operator overloading is used to overload operators to perform calculations.	2
4	Question / Answer • How many operators (unary & binary) are passed when we use friend function in operator overloading. • Which one is a better option ❖ To use a friend function ❖ To use a member function	5

Assignment to be given:-
1. Programs to implement the above concept.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 15

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:-IT-202 F

Subject: - OOPS Section - C

S. No.	**Topic: - Introduction to inheritance, base classes & derived classes, protected members.**	**Time Allotted:-**
1.	Introduction - Inheritance is a means of specifying hierarchical relationships between types - C++ classes can inherit both data and function members from other (parent) classes - Terminology: "the *child* (or *derived*) class *inherits* (or *is derived from*) the *parent* (or *base*) class."	10
2	Division of the Topic - Introduction to inheritance - Base classes & derived classes - Protected members.	5 5 5
3.	Conclusion Sometimes, however, we would like to be able to define encapsulated data members which are not publicly accessible, but which *are* accessible to derived classes.	10
4	Question / Answer ▪ Explain various types of inheritance? ▪ What are the applications of inheritance? ▪ Give any real time example for inheritance.	15

Assignment to be given:-
1. Different types of inheritance with block diagrams and examples.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 16

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section - C

S. No.	Topic: - Casting base classes & derived class pointers, using member functions.	Time Allotted:-
1.	Introduction Four new casting operators: - reinterpret_cast - dynamic_cast - static_cast - const_cast	10
2	Division of the Topic - Casting base classes & derived class pointers - using member functions.	15 5
3.	Conclusion Casts are used to convert the type of an object, expression, function argument, or return value to that of another type. Some conversions are performed automatically by the compiler without intervention by the programmer. These conversions are called *implicit conversion*s.	5
4	Question / Answer • What do you mean by type casting? • What is casting in case of classes? • What is the difference between simple casting & casting base classes & derived class pointers	10

Assignment to be given:-
1. Write a program to show casting of objects.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 17

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section - C

S. No.	**Topic :-** Overriding base class members in a derived class.	**Time Allotted:-**
1.	Introduction Overriding is used in case of inheritance. A derived class can inherit both public and protected members (both variable and functions) from a base class. However, the derived class can also redefine the inherited member function. If the derived class defines a member function with has the same signature (number and type of parameters) as the base class, then the derived class is ***overriding*** the base class's member function.	10
2	Division of the Topic Overriding base class members in a derived class.	15
3.	Conclusion Overriding is a case where we have same function name and same signatures. By signature, we mean to say that we have same number of parameters, same type of parameters and same return type.	10
4	Question / Answer • How overriding is different from overloading? • Can we use overriding & overloading with inheritance?	15

Assignment to be given:-

1. Write a program to show the concept of overriding and overloading.

Reference Readings:-

C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 18

Faculty:- Nidhi Jindal   Semester:-III   Class:- ECS   Course Code:- IT-202 F

Subject: - OOPS   Section - C

S. No.	**Topic :- Public , Protected & Private Inheritance, using constructors & destructors in derived classes**	**Time Allotted:-**
1.	Introduction - A member (either data member or member function) declared in a `private` section of a class can only be accessed by member functions and friends of that class - A member (either data member or member function) declared in a `protected` section of a class can only be accessed by member functions and friends of that class, and by member functions and friends of derived classes - A member (either data member or member function) declared in a `public` section of a class can be accessed by anyone	10
2	Division of the Topic - Public , Protected & Private Inheritance - using constructors & destructors in derived classes	15 10
3.	Conclusion Inheritance is what separates abstract data type (ADT) programming from OO programming.	5
4	Question / Answer • How do you express "`private` inheritance"? • How are "`private` inheritance" and "composition" similar? • Which should I prefer: composition or private inheritance?	10

Assignment to be given:-
1. Sequence in which the constructors are invoked in case of inheritance.
2. Base class initialization.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 19

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section - C

S. No.	**Topic: - Implicit derived class object to base class object conversion.** **Composition vs. Inheritance.**	**Time Allotted:-**
1.	Introduction The expression dynamic_cast<*type-id*> (expression) converts the operand *expression* to an object of type *type-id*. The *type-id* must be a pointer or a reference to a previously defined class type or a pointer to void. The type of *expression* must be a pointer if *type-id* is a pointer or an l-value if *type-id* is a reference.	10
2	Division of the Topic - Implicit derived class object to base class object conversion - Composition vs. Inheritance.	15 5
3.	Conclusion We have seen the conversion and also the differences between composition vs. inheritance.	10
4	Question / Answer • What is composition? • State some of the limitations of composition over inheritance.	5

Assignment to be given:-
1. Difference between composition and inheritance.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 20

Faculty:-Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section - C

S. No.	Topic :- Virtual functions & polymorphism, introduction to virtual functions, abstract base class & concrete classes	Time Allotted:-
1.	Introduction	
	Polymorphism refers to the capability to associate many meanings to one function name by means of a special mechanism known as a virtual function, or late binding.	
	At the design level, an abstract base class (ABC) corresponds to an abstract concept.	10
	At the programming language level, an ABC is a `class` that has one or more pure virtual member functions. You cannot make an object (instance) of an ABC.	
2	Division of the Topic	
	- Virtual functions & polymorphism	5
	- Introduction to virtual functions	5
	- Abstract base class & concrete classes	5
3.	Conclusion	
	Run time polymorphism, virtual functions and dynamic binding are terms having similar meaning.	10
4	Question / Answer	
	• What is a "`virtual` member function"?	
	• How can C++ achieve dynamic binding yet also static typing?	15

Assignment to be given:-
1. What are abstract classes? Where are they used?

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 21

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section - C

S. No.	**Topic: - Polymorphism, new classes & dynamic binding, virtual destructors.**	**Time Allotted:-**
1.	Introduction *Dynamic binding* means that the address of the code in a member function invocation is determined at the last possible moment: based on the dynamic type of the object at run time. It is called "dynamic binding" because the binding to the code that actually gets called is accomplished dynamically (at run time). Dynamic binding is a result of `virtual` functions.	10
2	Division of the Topic - Polymorphism - New classes & dynamic binding - Virtual destructors.	 5 5 5
3.	Conclusion You need a base class's destructor to be `virtual` if and only if you intend to allow someone to invoke an object's destructor via a base class pointer and the object being destructed is of a derived class that has a non-trivial destructor.	10
4	Question / Answer • What's the difference between how `virtual` and non-`virtual` member functions are called? • What happens in the hardware when I call a virtual function? How many layers of indirection are there? How much overhead is there? • How can a member function in my derived class call the same function from its base class?	15

Assignment to be given:-
1. Program to show the implementation of dynamic binding.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 22

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section – D

S. No.	Topic: - Files & streams, creating a sequential access file, reading data from a sequential access file, updating sequential access file.	Time Allotted:-
1.	Introduction There are 3 File I/O classes in C++ which are used for File Read/Write operations. They are - ifstream - Can be used for File read/input operations - ofstream - Can be used for File write/output operations - fstream - Can be used for both read/write c++ file I/O operations Sequential Access Files: These files must be accessed in the same order in which they were written. This process is analogous to audio cassette tapes where you must fast forward or rewind through the songs sequentially to get to a specific song.	10
2	Division of the Topic - Files & streams - Creating a sequential access file - Reading data from a sequential access file - Updating sequential access file.	5 5 5 5
3.	Conclusion So, the most important methods which will be used for any file operations are: - fstream::open method - to open the file - fstream::Operator >> and fstream::Operator << - For reading from or writing to the file. - fstream::close - Flushes all buffer and close the file	5
4	Question / Answer • What is the difference between standard file and binary file. Which one is better?	10

Assignment to be given:-
1. Different commands for reading and writing data into/from a file incase of character file and binary file.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 23

Faculty:-Nidhi Jindal Semester:-III Class:- ECs Course Code:- IT-202 F

Subject: - OOPS Section – D

S. No.	**Topic: - Random access files, creating a random access file, writing data randomly to a random access file, reading data sequentially from a random access file.**	**Time Allotted:-**
1.	Introduction Random Access Files: These files are analogous to audio compact disks where you can easily access any song, regardless of the order in which the songs were recorded. Random access files allow instant access to any data in the file. Unfortunately, random access files often occupy more disk space than sequential access files.	10
2	Division of the Topic - Random access files - Creating a random access file - Writing data randomly to a random access file - Reading data sequentially from a random access file.	5 5 5 5
3.	Conclusion Files are the medium for storage.	5
4	Question / Answer • Difference between random access and sequential access file? • Why we need files?	15

Assignment to be given:-
1. Write a program to read the student information and storing it in a sequential file.
Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 24

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS Section – D

S. No.	**Topic: - Stream I/O classes & objects, stream I/O, unformatted I/O, stream manipulators, stream format states, stream error states.**	**Time Allotted:-**
1.	Introduction • C++ provides various *stream manipulators* that perform formatting tasks. • Stream manipulators are defined in <iomanip> • These manipulators provide capabilities for – setting field widths, – setting precision, – setting and unsetting format flags, – flushing streams, – inserting a "newline" and flushing output stream, – skipping whitespace in input stream	10
2	Division of the Topic	
	- Stream I/O classes & objects	5
	- Stream I/O, unformatted I/O	5
	- Stream manipulators	5
	- Stream format states,	5
	- Stream error states.	5
3.	Conclusion Manipulators are used in case of formatting our output on the display device.	5
4	Question / Answer • How many types of manipulators are there? • Explain the function of fill() function?	10

Assignment to be given:-
1. Different types of manipulators with the example of their usage.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 25

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section – D

S. No.	**Topic: - Function templates, overloading template functions, class templates, non-type parameters.**	**Time Allotted:-**
1.	Introduction C++ templates allow one to implement a generic Queue<T> template that has a type parameter T. T can be replaced with actual types, for example, Queue<Customers>, and C++ will generate the class Queue<Customers>. Changing the implementation of the Queue becomes relatively simple. Once the changes are implemented in the template Queue<T>, they are immediately reflected in the classes Queue<Customers>, Queue<Messages>, and Queue<Orders>. C++ provides two kinds of templates: class templates and function templates.	10
2	Division of the Topic - Function templates - Overloading template functions - Class templates - Non-type parameters.	5 5 5 5
3.	Conclusion Templates are very useful when implementing generic constructs like vectors, stacks, lists, queues which can be used with any arbitrary type. C++ templates provide a way to re-use source code as opposed to inheritance and composition which provide a way to re-use object code.	5
4	Question / Answer • What's the idea behind templates? • What's the syntax / semantics for a "class template"? • What's the syntax / semantics for a "function template"? • How do I explicitly select which version of a function template should get called? • What is a "parameterized type"? • What is "genericity"?	15

Assignment to be given:-
1. Example of class template and function template.
Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 26

Faculty:- Nidhi Jindal Semester:-III Class:- ECS Course Code:- IT-202 F

Subject: - OOPS . Section – D

S. No.	**Topic :-** Template & inheritance, templates & friends, templates & static members.	**Time Allotted:-**
1.	Introduction Templates are related to inheritance; even they can be used with friend functions.	10
2	Division of the Topic - Template & inheritance - Templates & friends - Templates & static members.	 5 5 5
3.	Conclusion This lecture was to define the relationship between template and inheritance.	10
4	Question / Answer • Explain the significance of using template with inheritance, templates with friends and templates with static members.	15

Assignment to be given:-
1. Program to show the concept of Templates.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Lecture Plan 27

Faculty:-Nidhi Jindal Semester:-III Class:- ECs Course Code:- IT-202 F

Subject: - OOPS Section – D

S. No.	**Topic: -** Introduction to exception handling: try, throw, catch, throwing an exception, catching an exception, re-throwing an exception, exception specifications.	**Time Allotted:-**
1.	Introduction Exception handling is all about how to avoid unexpected conditions. There are three keywords for it: - try - catch - throw	5
2	Division of the Topic - Introduction to exception handling: try, throw, catch - Throwing an exception, - Catching an exception, - Re-throwing an exception, - Exception specifications.	5 5 5 5 5
3.	Conclusion This lecture gave a good idea about exception handling.	5
4	Question / Answer • What are some ways `try` / `catch` / `throw` can improve software quality? • How can I handle a constructor that fails? • How can I handle a destructor that fails? • How should I handle resources if my constructors may throw exceptions? • How do I change the string-length of an array of `char` to prevent memory leaks even if/when someone throws an exception? • What should I throw? • What should I catch?	15

Assignment to be given:-

1. How do I change the string-length of an array of `char` to prevent memory leaks even if/when someone throws an exception?

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy

Doc. No.: DCE/0/15
Revision: 00

Lecture Plan 28

Faculty:- Nidhi jindal Semester:-III Class:- ECs Course Code:- IT-202 F

Subject: - OOPS Section – D

S. No.	**Topic: -** Processing unexpected exception, stack unwinding constructors, destructors, inheritance & exception handling.	**Time Allotted:-**
1.	Introduction Introduction was all about processing unexpected exception, stack unwinding constructors, destructors, inheritance & exception handling.	10
2	Division of the Topic - Processing unexpected exception - Stack unwinding constructors - Destructors - Inheritance & exception handling.	 5 5 5 5
3.	Conclusion This lecture gave a good idea about processing unexpected exception, stack unwinding constructors, destructors, inheritance & exception handling.	10
4	Question / Answer • How exception handling can be done in case of inheritance	5

Assignment to be given:-
1. Program to show the above concept.

Reference Readings:-
C++How to Program by Dietel and Dietel
Object Oriented Programming by Balagurusamy