Object Oriented programming [OOPS]

* Assignment 1 *

1] What do you mean by object oriented programming?
Ans Object oriented programming or OOPs refers to languages that use objects in programming. Object-oriented programming aims to implement real-world entities like inheritance, polymorphism, etc in programming. The main aim of OOPS is to bind together the data & the functions that operate on them so that no other part of the code can access this data except that function.

2] What do you mean by structure of a program?
Ans Structure of a program refers to a sequence of instructions or statements. These statements are what form the structure of a program. The program structure divides into several sections which are namely headers, class definition, etc.

3] What are paradgims?
Ans Paradgims can also be termed as method to solve some problem or do some task. There are lots of programming language that are known but all of them need to follow some strategy when they are implemented & this methodology/strategy is paradigms. Apart from variety of programming language there are lots of paradigms to language fulfill each & every demand.

4] What do you mean by variable?
Ans Variable are names given to computer memory locations in order to store data in a program. This data can be known or unknown based on the assignment of value to the variables. Variables can also be considered as

'containers' which are used to hold more than one value.

5.] What are arrays?

Ans An array is a group of similar elements or data items of the same type collected at contiguous memory locations. In other words, we can say that in computer programming, arrays are generally used to organize the same type of data.

6.] Define & explain different paradigms for oops.

Ans There are 5 types of paradigms -

1.] Inheritance - It is a mechanism that permits new classes to be created out of existing classes by extending & refining its capabilities. The existing classes are called the base classes / parent classes & the new classes are called the child class.
The subclass can inherit or derive the attributes & methods of the super-class provided that the super-class allow so, the subclass may add its own attributes & methodes.

2.] Encapsulation - Encapsulation is the process of binding both attributes & methods together within a class. Through encapsulation, the internal details of a class can be hidden from outside. It permits the elements of the class to be accessed from outside only through the interface provided by the class.

3.] Polymorphism - Polymorphism is originally a greek word that means the ability to take multiple forms. In oops paradigms, polymorphism implies using operations in different ways, depending upon the instances they are operating upon. Polymorphism allows object with different internal structures to have a common external interface.

4.] Composition - Composition is a relationship among classes by which a class can be made up of any combination of objects of other classes. It allow objects to be placed directly within the body of other classes.

5.] Interface - Interface in oops is an programming structure / syntax that allows the computer to enforce certain properties on an object [class]. It also refers to as universal method. It describes a set of method signatures, the implementations of which may be provided by multiple classes.

7.] Explain the concept of oops [object, class, inheritance, ---etc] with real ex-?

Ans The concepts of oops are as follows-

1.] class- A class is a data type with data type members and member functions that is defined by the user. Data members are data variables, & members functions are functions that manipulate these variables; these data members & member functions together define the properties & behaviour of the objects in a class.

Real time example- If you had a class called "Expensive cars" it could contain objects like Mercedes, BMW, Toyota, an so on. The price or speed of these autos could be one of its attributes (data). Driving, booking & other technique can be used with these vehicles.

2.] object - An object is an instance of a class. When a class is defined, no memory is allocated but it is instantiated i.e, an object is created memory is allocated.

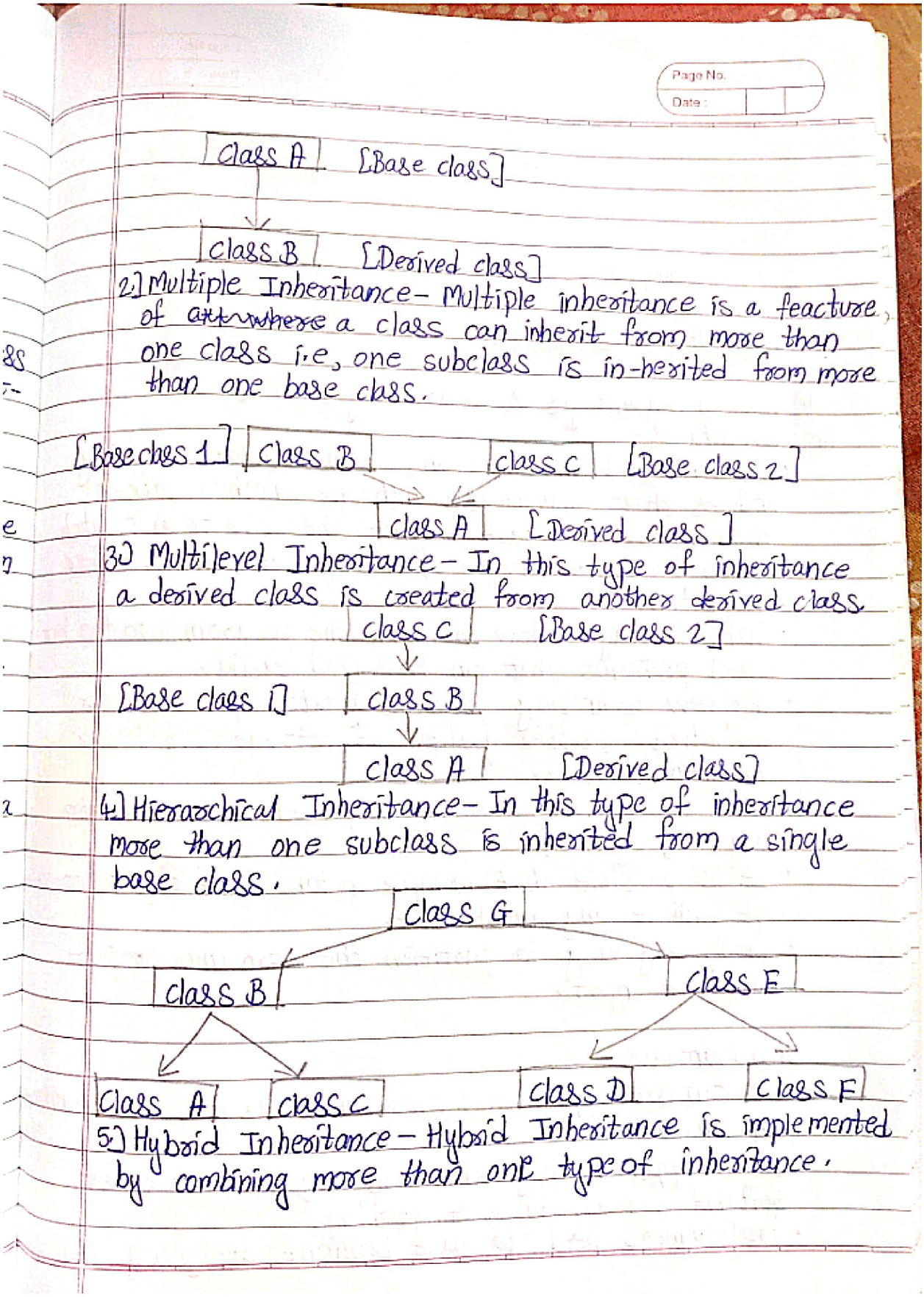
Real time ex- The objects of a class called animals for example will be a cat, dog, cow & so on. Each object

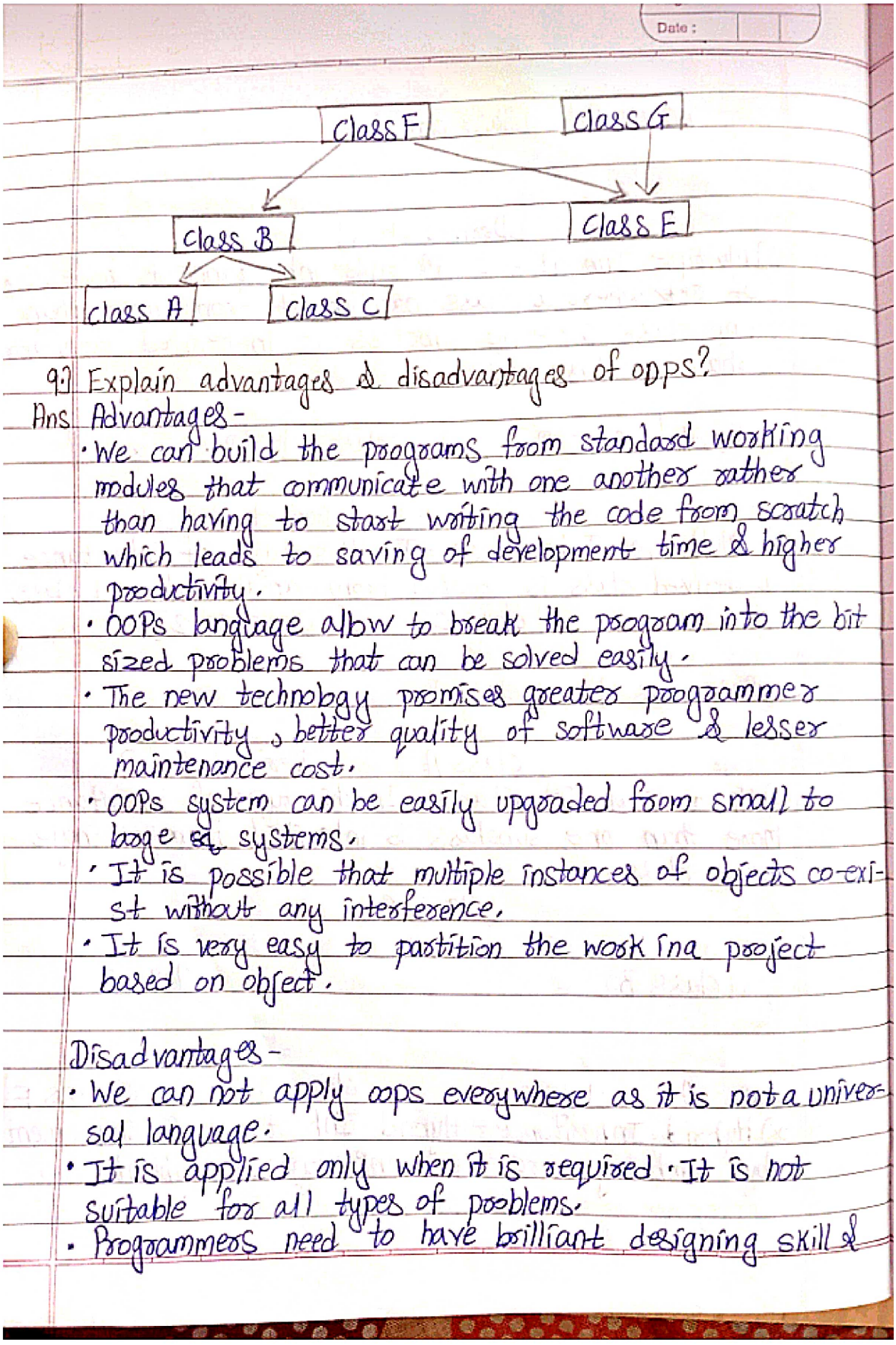
has its own identity attribute & behaviour. Ex:

3.] Inheritance - classes can be grouped into hierarchies with each parent or child class having one or more children. If a class has a parent class, it is said to be derived or inherited from it.
Ex- An insect could be represented by an insect superclass in the animal world. Insects all have the same characteristics such as six legs & an exoskeleton.

4.] Encapsulation - In encapsulation data & methods that operate on it by combining them into a single unit, the class. We can hide private information of a class from the outside world & only disclose functionality that is needed to interact with it this way
Ex- Extending the person class example from before, the class may contain private data such as "social security Number" that should not be disclosed to other program objects. Outside programmes would not have direct access to this data member beause it was encapsulated as a private variable in the class.

8.] Explain inheritance & its types and ex?
Ans Inheritance - It is a mechanism that permits new classes to be created out of existing classes by extending & refining its capabilities. The existing classes are called the base classes or parent class.
Types of inheritance are as follows -
1.] Single Inheritance - In single inheritance, a class is allowed to inherit from only one class. i.e one sub-class is inherited by one base class only.


Page No.
Date :
Class A [Base class]
Class B [Derived class]
2.] Multiple Inheritance - Multiple inheritance is a feacture of ~~anywhere~~ a class can inherit from more than one class i.e, one subclass is in-herited from more than one base class.
[Base class 1]
Class B
class C
[Base class 2]
class A [Derived class]
3.) Multilevel Inheritance - In this type of inheritance a derived class is created from another derived class.
class C [Base class 2]
[Base class 1]
class B
class A [Derived class]
4.] Hierarchical Inheritance - In this type of inheritance more than one subclass is inherited from a single base class.
class G
class B
class E
Class A
class C
class D
class F
5.] Hybrid Inheritance - Hybrid Inheritance is implemented by combining more than one type of inheritance.



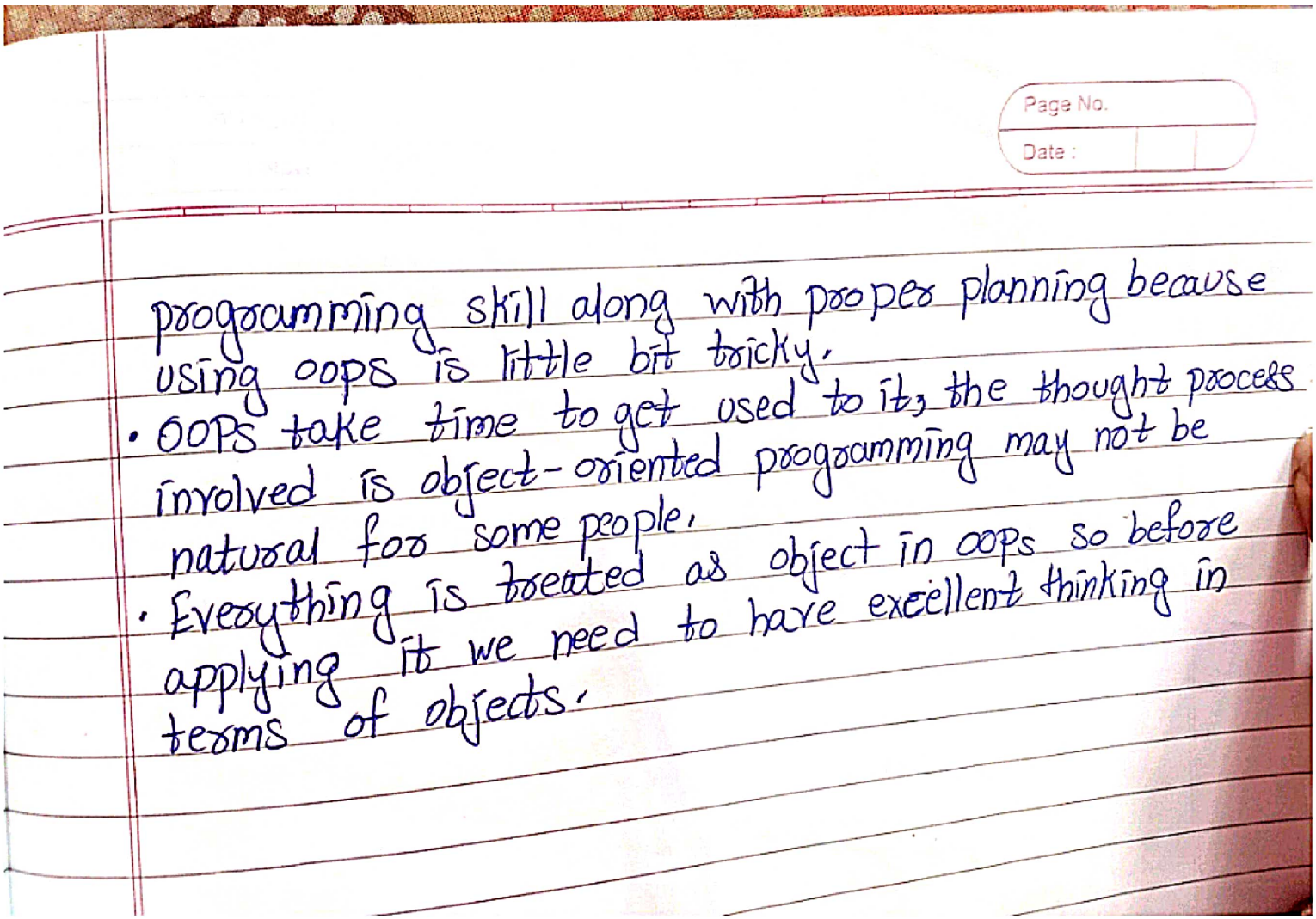
Q.) Explain advantages & disadvantages of opps?

Ans. Advantages -

- We can build the programs from standard working modules that communicate with one another rather than having to start writing the code from scratch which leads to saving of development time & higher productivity.
- OOPs language allow to break the program into the bit sized problems that can be solved easily.
- The new technology promises greater programmer productivity, better quality of software & lesser maintenance cost.
- OOPs system can be easily upgraded from small to large systems.
- It is possible that multiple instances of objects co-exist without any interference.
- It is very easy to partition the work in a project based on object.

Disadvantages -

- We can not apply oops everywhere as it is not a universal language.
- It is applied only when it is required. It is not suitable for all types of problems.
- Programmers need to have brilliant designing skill &



programming skill along with proper planning because using oops is little bit tricky.

- OOPS take time to get used to it, the thought process involved is object-oriented programming may not be natural for some people.
- Everything is treated as object in oops so before applying it we need to have excellent thinking in terms of objects.

* Assignment - 2 *

1] What are data types?

Ans All variables use data type during declaration to restrict the type of data, to be store. Therefore we can say that data types are used to tell the variables the type of data they can store. Whenever a variable is defined in c++, the complier allocates some memory for that variable basedon the data type with which it's declared. Every data type requires a different amount of memory.

2] What are operators?

Ans An operator is a symbol that operates on a value to perform specific mathematical or logical computations. They form the foundation of any programming language. In c++, we have built-in operators to provide the required functionality.

An operator operates the operands. For example-

```
int c = a + b;
```

Here, '+' is the addition operator. 'a' & 'b' are the operands that are being added.

3] What do you mean by evaluation of expression?

Ans Evaluate an expression represented by a string. The expression can contain parantheses, you can assume parentheses are well-matched. For simplicity, you can assume only binary operations allowed are +, -, *, & /. Arithmetic expressions can be written in one of three forms.

- Infix Notation - Operators are written between the operands they operate on, eg. 3+4.
- Prefix Notation - Operators are written before the operands eg. +34
- Postfix Notation - Operators are written after operands.

4.] What are pointers?
Ans Pointers are the variables that stores the value of another variable. It enables us to work directly with memory location. It requires careful handling to avoid the error and security vulnerabilities. It provides a way to access and manipulate the data indirectly.

5.] What are flow control statements?
Ans In computer science, control flow is the order in which individual statements, instructions or function calls of an program are executed or evaluated. It redirects the flow of a program in order to execute add final code.
There are various flow control statements-
If statement, If-else statement, switch statement, for loop, while loop, do-while loop, break statement, continue statement, goto statement

6.] Explain the structure of C++ program in detail.
Ans
```
#include <iostream>  --> Header file
using namespace std; --> Standard Name space
int main () {        --> Main function
  int num 1= 24;     } --> Variable declaration
  int num 2= 36;     }
  int result=num 1+num2; --> Expression
cout << result <<endl; --> output
  return 0;          --> Result statement
}
```

- Header file- Generally, a program includes various programming elements like built-in functions, classes, keywords, constants, etc that are already defined in the standard

C++ library - Standard header are specified in a program through the preprocessor directive #include. When the complier processes the instruction #include<iostream>, it includes the contents of the stream in the program.

- Standard Namespace - The namespace permits grouping of various entities like classes, objects, functions & various C++ tokens, etc under a single name. Any user can create separate namespaces of its own & can use them in any other program.
- Main function - The main function tells the complier where to start the execution of the program. The execution of the program starts with the main function. The complier executes all the instructions which are written in the curly braces{} which encloses the body of the main function.
- Variable declaration - It is used to declare some constants & assign them some value. The variables and the class definitions which are going to be used in the program are declared to make them global. The scope of the variable declared in this section lasts until the entire program terminates.
- Expression - An expression is a valid arrangement of variables, constants and operators. In C++ program each expression is evaluated to compute a value of a given type.
- Output - It is used to display the output to the standard output device. The data needed to be displayed on the screen is inserted in the standard output stream (cout) using the insertion operator (<<).
- Result statement - The statement ends the execution of a function & returns control to the calling function. A return statement can return a value to the calling function.

Page No.
Date

Q] Explain different types of operators with example in detail.
Ans- Types of operators are as follows-
1] Arithmetic operator - These operators are used to perform mathematical operation on operands. ex-[+, -, *, /, %, ++, --]
Arithmetic operators are of two types-
A] Unary operator- Operators that operate or work with a single operand are unary operators. Ex - Increment [++] & Decrement [--] operators.

```
int val = 5;
cout<<++val; //6
```

B] Binary Operators - Operators that operate or work with 2 operands are binary operators. Ex - Addition [+], Substraction [-], Multiplication [*], Division [/] operators.

```
int a=7;
int b=2;
cout<<a+b; //9
```

2] Relational operators - These are used for the comparison of the values of two operands. For ex - checking if one operand is equal to the other operand or not, whether an operand is greater than the other operand or not, etc. Some of the relational operators are [==, >=, <=]

```
int a=3;
int b=5;
cout << (a<b);
//operator to check if [a] is smaller than [b].
```

3] Logical operators - It is used to combining two or more constraints or to complement the evaluation of the original condition in consideration. The result of the operation of a logical operator is a boolean value either true or false. For ex - the logical AND represented as the '&&' operator in C returns true when both the conditions under consideration are satisfied.

```
cout <<(4 != 5) && (4<5); //true
```

4] Bit wise Operator - It is used to perform bit-level operations on the operands. The operators are first converted to bit-level & then the calculation is performed on the operand. Mathematical operations can be performed at the bit level for faster processing. For ex - the AND operator represented as '&' in C takes 2 numbers as operands & does AND on every bit of 2 numbers.

```
int a=5, b=9; // a=
5(00000101), b=9(00001001)
cout << (a^b); //
00001100
cout << (~a); //
11111010
```

5] Assignment operator - It is used to assign value to a variable. The left side operand of the assignment operator is a variable & the right side operand of the assignment operator is a value. The value on the right side must be of the same data type as the variable on the left side the compiler will raise an error.

Ex - '=' - It is used to assign value to right to the variable on the left,

```
a=10;
b=20;
ch='y';
```

"+=" - This operator first adds the current value of the variable on left to the value on the right then assigns the result to the variable on the left.

(a+=b) can be written as

(a=a+b)

If initially value stored in a is 5. Then (a+=6)=11

8] Explain different types flow control statements in detail.

Ans. Flow control statements - A C++ control statement redirects the flow of a program in order to execute add final code.

1] if statement - It is used to test a condition & execute a block of code if the condition is true

Syntax - if (condition) {
    // statement to execute if
    // condition is true
}

2] if else statement - if condition tell us that if a condition is true it will execute a block of statement & if the condition is false it won't. But what if it want to do something else if the condition is false, we can use else statement with if statement to ~~ensure~~ execute a block of code if condition is false.

Syntax - if (condition) {
    // execute this block if
    // condition is true
}
else
{
    // execute this block if
    // condition is false
}

3] switch statement - It is a flow control statement that is used to execute the different blocks of statement based on the value, the given expressions. We can create diff^{n} cases for diff^{n} values of the switch expression.

Syntax - switch (expression) {
    case value 1:
        // statement 1
        break;

Date :

```
case value 2:
  //statement 2
  break;

case value 3:
  //statement 3
  break;
  :
  default:
    //default statement
    break;
}
```

4] for loop - for loop is used to execute a code specified number of times.

Syntax - for (initialization; condition; increment / decrement)
{
  //code to execute specific time
}

5] While loop - It is used to execute a block of code as long as condition.

Syntax - While condition{
  //code to execute ass long as
  // condition is true
}

6] do while loop - It is used to execute a block of code atleast once & then continue executing the block of code as long as condition is true.

Syntax - do{
  //code to execute atleast once
}
while (condition);

7] break statement - The breaking c++ loop control statement that is used to terminate the loop. As soon as the break statement is encountered from within a the loop.
Syntax - break;

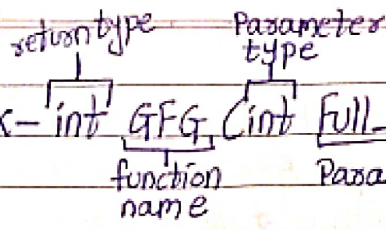
8] continue statement - It is a loop control statement that forces the program control to execute the next iteration of the loop. As a result, the code inside the loop following the continue statement will be skipped as the next iteration loop will begin.
syntax - continue;

9] goto statement - The goto statement is a loop statement which is sometimes also referred to a unconditional jump statement. The goto statement can be used to jump from anywhere to anywhere within a function.
syntax -

syntax 1	syntax 2
goto label;	label;
label;	goto label;

9] Explain functions, parameters passing & scope of variables in detail?
Ans A function is a set of statements that takes input does some specific computation and produces output. The idea is to put some commonly or repeatedly done tasks together to make a function so that instead of writing the same code again & again for different inputs, we can call this function. In simple words, a funtion is a block of code that runs only when it is called.

Page No.
Date :



Functions help us in reducing code redundancy. If functionality is performed at multiple places in software, then rather than writing the same code, again & again, we create a function & call it every where. This also helps in maintenance as we have to make changes in only one place if we make changes to the functionality in future.

- Parameter passing

There are different ways in which parameter data can be passed into & out of methods & functions. Let us assume that a function B() is called from another function A(). In this case, A is called the "caller function" & B is called the "called function or callee function". Also, the arguments which A sends to B are called actual arguments and the parameters of B are called formal arguments.

Basic terminologies-

- function Parameters - These variables, which indicate the data that the function anticipates receiving when called are specified in the parameter list of a function.
- Actual parameters - The expressions or values passed in during a function call. When the function is called, it receives these values as inpt.

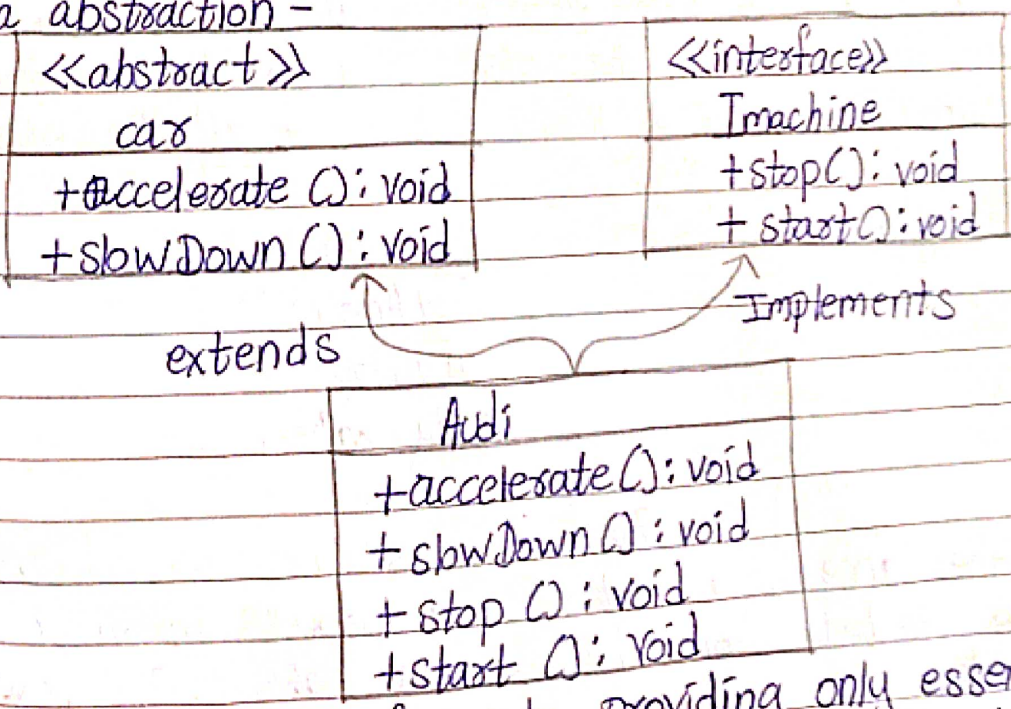
The parameter that the function signature declares. They serve as stand-ins for the values that the are provided in when the function is called.

. Scope of variables -
In general, the scope is defined as the extent up to which something can be worked with. In programming also the scope of a variable is defined as the extent of the program code within which the variable can be accessed or declared or worked with. There are mainly 2 types of variable scopes.

1) Local variable - Local variables do not exist outside the block in which they are declared. i.e, they can not be accessed or used outside that block.
2) Global variables - Global variables are usually declared outside of all of the functions & blocks at the top of the program. They can be accessed from any position of the program.

12] Explain data abstraction, encapsulation, inheritance & polymorphism in detail with diagram & ex-

Ans Data abstraction -



Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.

Page No.
Date :

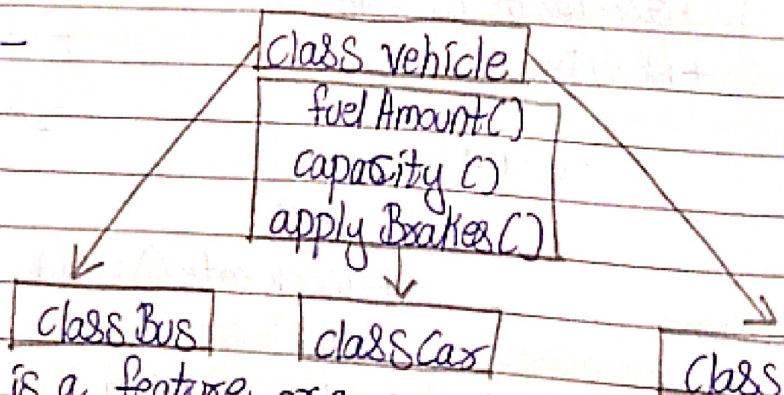
Real time exp. example- The man only knows that pressing the accelerator will increase the speed of the car or applying brakes will stop the car but he does not know how on pressing the accelerator the speed is actually increasing, he does not know about the inner mechanism of the car or the implementation of the accelerator, brakes, etc in the car. This is what abstraction is.

Encapsulation- class { Methods | Variables }

Encapsulation in c++ is defined as the wrapping up of data & information in a single unit. In oops encapsulation is defined as binding together the data & the functions that manipulate them.

Consider a real-life ex of encapsulation, in a company, there are diff^n sections like the accounts section, finance section, sales section, etc. Now, the finance handles all the financial transactions & keep records of all the data related to finance. Similarly, the sales section handles all the sales-related activities & keeps records of all the sales.

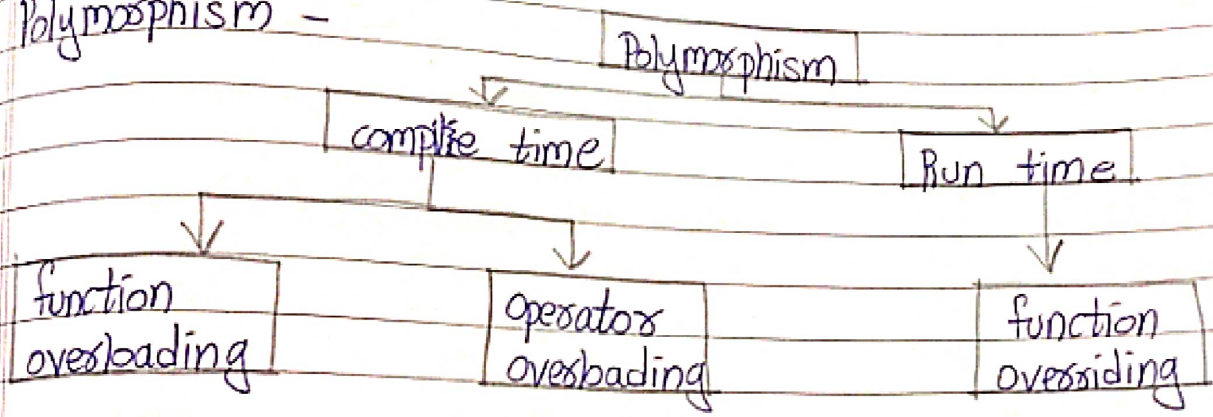
Inheritance -



Inheritance is a feature or a process in which new classes are created from the existing classes. The new class created is called derived class or child class & the existing class is known as the base class or parent class.

ex- An insect could be represented by an insect superclass in the animal world. Insects all have the same characteristics such as six legs & an exoskleton.

Polymorphism –



Polymorphism is the ability of any data to be processed in more than one form. The most common use of polymorphism in oops occurs when a parent class reference is used to refer to a child class object.

Ex- A person at the same time can have different roles to play in life. So that the same person has to have many features but has to implement each as per the situation & the condition.

* Assignment 3 *

1] What are classes in oops?
Ans The building block of c++ that leads to Object Oriented programming is a class. It is a User defined data type, which holds its own data members & member functions, which can be accessed & used by creating an instance of that class. A class is like a blueprint for an object. A class is a user-defined data type that has data members & member functions.

2] What do you mean by data abstraction?
Ans Data abstraction is one of the most essential & important features of oops in c++. Abstraction means displaying only essential information & hiding the details. Abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation. We can implement abstraction in c++ using classes.

3] What are constructors?
Ans They are used to construct a new class or objects. Constructors are special methods in oops language like c++, java & python. These play fundamental role in initialization of objects. Constructors are used to provide default values for object. You can have multiple constructors in c++, java in a class. The main purpose of constructors is object initialization.

4] What are functions?
Ans Functions is a set of statements that takes input, does some specific computation & produces output. The idea is to put some commonly or repeadly done tasks

together to make a function so that instead of writing the same code again and again for different inputs we call this function. Function help use in reducing code redundancy.

5] What are destructors?
Ans Destructors are the concept or method in oops which are used to kill or disturb the object in the class. They are special methods which are used to cleanup & release the resources that the object might have required. Destructors are used to perform tasks- 1] cleaning up resources 2] Releasing memory 3] Maintaining object Integrity.

6] What is data abstraction discuss in detail & give its advantages and disadvantages?
Ans Data abstraction is one of the most essential and important features of object oriented programming in c++. Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation
Types of Abstraction-
1] Data abstraction- This type only shows the required information about the data and hides the unnecessary data.
2] control Abstraction-

Advantages of data abstraction-
- Helps the user to avoid writing the low-level code.
- Avoids code duplication & increases reusability.
- Can change the internal implementation of the class independently without affecting the user.

- Helps to increase the security of an application or program as only important details are provided to the user.
- It reduces the complexity as well as the redundancy of the code, therefore increasing the readability.

Disadvantages of data abstraction

- For executing an abstraction, the code implementation must handle cases & situations which is not always necessary or often aren't needed - by many usage scenarios.
- This will make the code slow in comparison to the code which is directly implemented in the operation without using the abstraction.
- This does not matter on larger systems in today's world, but it may create problems in small devices or constrained environments.

7.] Describe the role of constructors & destructors and types also.

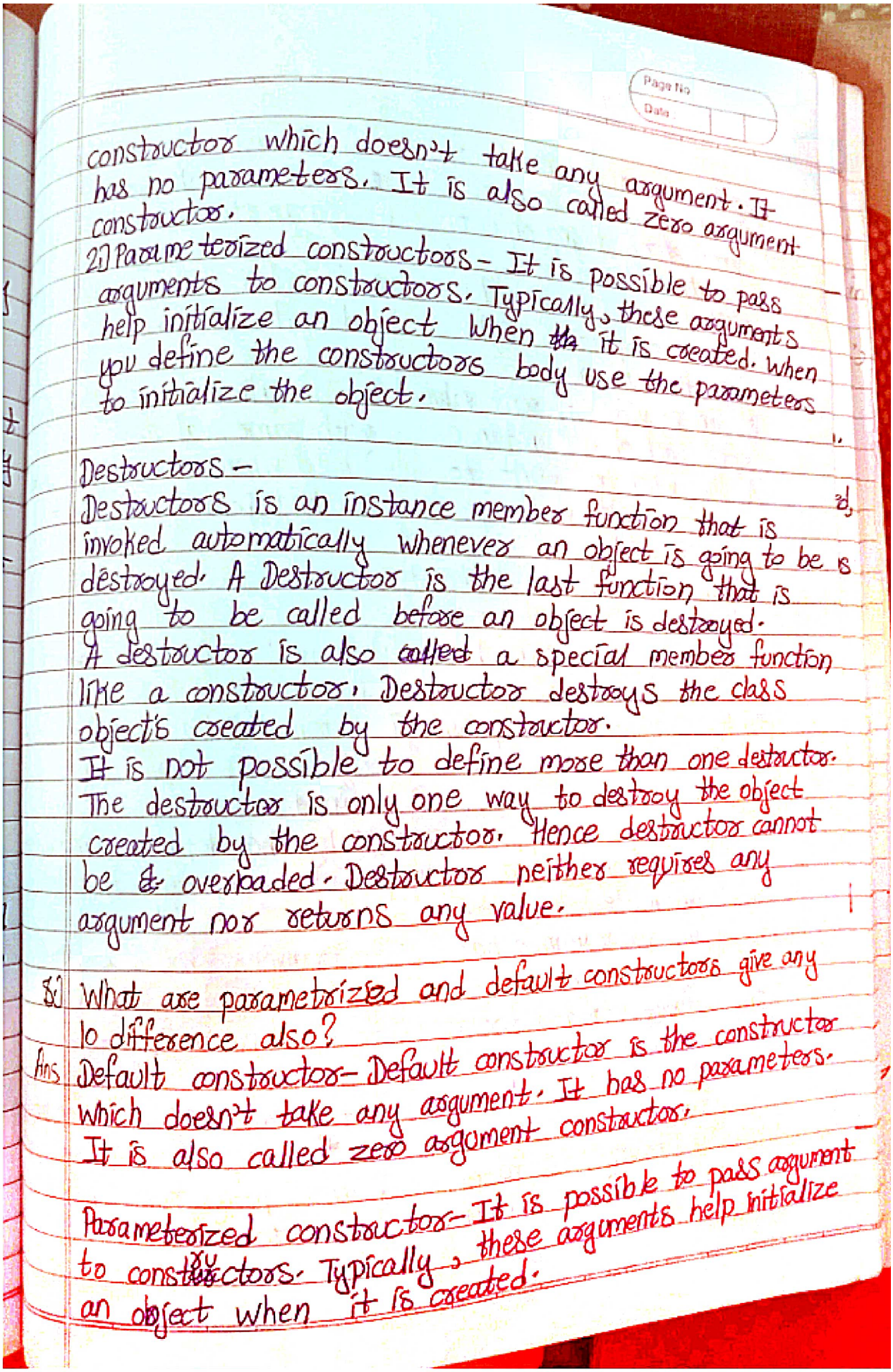
Ans Constructors -

Constructors in C++ is a special method that is invoked automatically at the time of object creation. It is used to initialize the data members of new objects generally. The main purpose of the constructor are -

Constructor often have the same name as the declaring class. They have the task of initializing the objects data members and of establishing the invariant of the class, failing in the invariant is invalid. A properly written constructor leaves the resulting object in a valid state.

Types of constructors -

1) Default constructors - Default constructors is the



constructor which doesn't take any argument. It has no parameters. It is also called zero argument constructor.

2) Parameterized constructors - It is possible to pass arguments to constructors. Typically, these arguments help initialize an object when it is created. When you define the constructors body use the parameters to initialize the object.

Destructors -

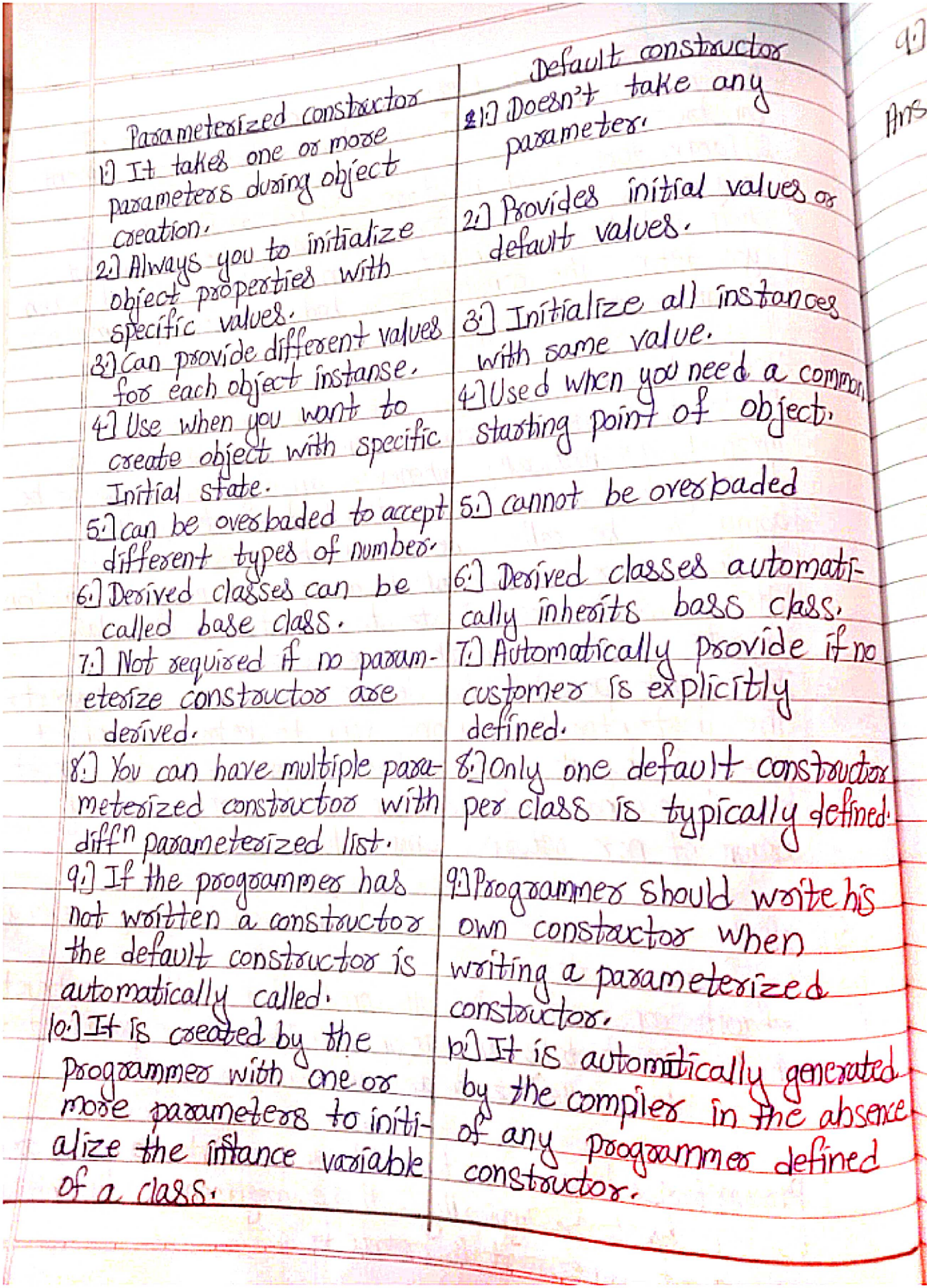
Destructors is an instance member function that is invoked automatically whenever an object is going to be destroyed. A Destructor is the last function that is going to be called before an object is destroyed. A destructor is also called a special member function like a constructor. Destructor destroys the class object's created by the constructor.

It is not possible to define more than one destructor. The destructor is only one way to destroy the object created by the constructor. Hence destructor cannot be overloaded. Destructors neither requires any argument nor returns any value.

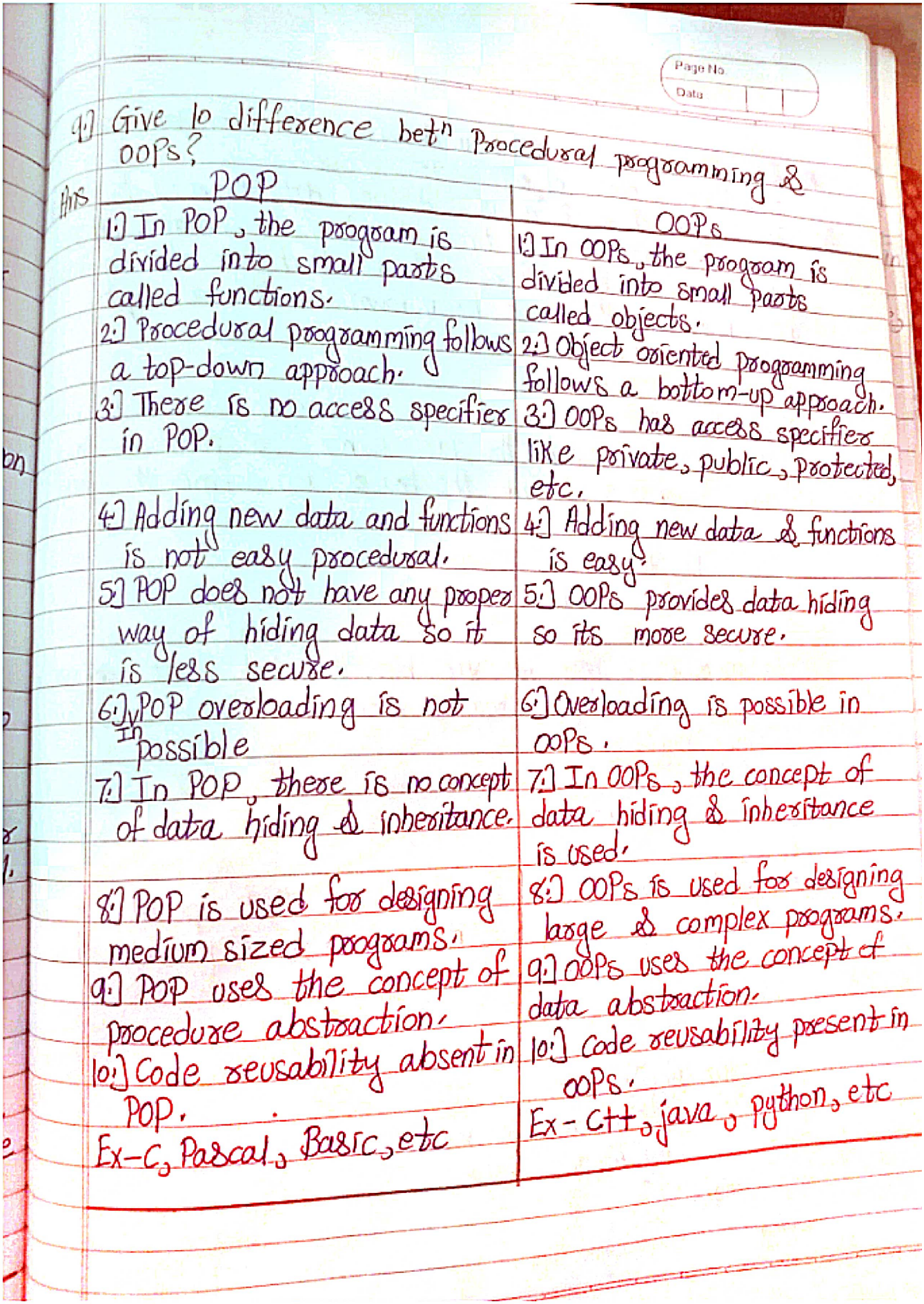
Q) What are parametrized and default constructors give any to difference also?

Ans) Default constructor - Default constructor is the constructor which doesn't take any argument. It has no parameters. It is also called zero argument constructor.

Parameterized constructor - It is possible to pass argument to constructors. Typically, these arguments help initialize an object when it is created.



Parameterized constructor	Default constructor
1.] It takes one or more parameters during object creation.	1.] Doesn't take any parameter.
2.] Always you to initialize object properties with specific values.	2.] Provides initial values or default values.
3.] Can provide different values for each object instanse.	3.] Initialize all instances with same value.
4.] Use when you want to create object with specific Initial state.	4.] Used when you need a common starting point of object.
5.] can be overbaded to accept different types of numbers.	5.] cannot be overbaded
6.] Derived classes can be called base class.	6.] Derived classes automatically inherits bass class.
7.] Not required if no parameterize constructor are derived.	7.] Automatically provide if no customer is explicitly defined.
8.] You can have multiple parameterized constructor with diffn parameterized list.	8.] Only one default constructor per class is typically defined.
9.] If the programmer has not written a constructor the default constructor is automatically called.	9.] Programmer should write his own constructor when writing a parameterized constructor.
10.] It is created by the programmer with one or more parameters to initialize the instance variable of a class.	b.] It is automatically generated by the compiler in the absence of any programmer defined constructor.



1.] Give 10 difference betn Procedural programming & OOPs?

POP	OOPs
1.] In POP, the program is divided into small parts called functions.	1.] In OOPs, the program is divided into small parts called objects.
2.] Procedural programming follows a top-down approach.	2.] Object oriented programming follows a bottom-up approach.
3.] There is no access specifier in POP.	3.] OOPs has access specifier like private, public, protected, etc.
4.] Adding new data and functions is not easy procedural.	4.] Adding new data & functions is easy.
5.] POP does not have any proper way of hiding data so it is less secure.	5.] OOPs provides data hiding so its more secure.
6.] POP overloading is not possible	6.] Overloading is possible in OOPs.
7.] In POP, there is no concept of data hiding & inheritance.	7.] In OOPs, the concept of data hiding & inheritance is used.
8.] POP is used for designing medium sized programs.	8.] OOPs is used for designing large & complex programs.
9.] POP uses the concept of procedure abstraction.	9.] OOPs uses the concept of data abstraction.
10.] Code reusability absent in POP.	10.] Code reusability present in OOPs.
Ex- C, Pascal, Basic, etc	Ex- C++, java, Python, etc

10) Describe the c++ classes in detail & also explain class structure, class objects, class scope.

Ans In c++, a class is a user defined data type that serves as a blueprint for creating objects. It encapsulates data members & member functions that operate on those data members. Here's a breakdown of key concepts related to c++ classes:

1) Class structure-

Declaration- The class is declared using the 'class' keyword, followed by the class name & a pair of curly braces. Inside the braces, you define the class members.

```
class MyClass {
    // class members go here
};
```

Data members - These are variables that hold data for each object created from the class.

```
class MyClass {
public:
    int myInteger;
};
```

Member functions - These are functions that operate on the class's data members & provide the behavior of the class.

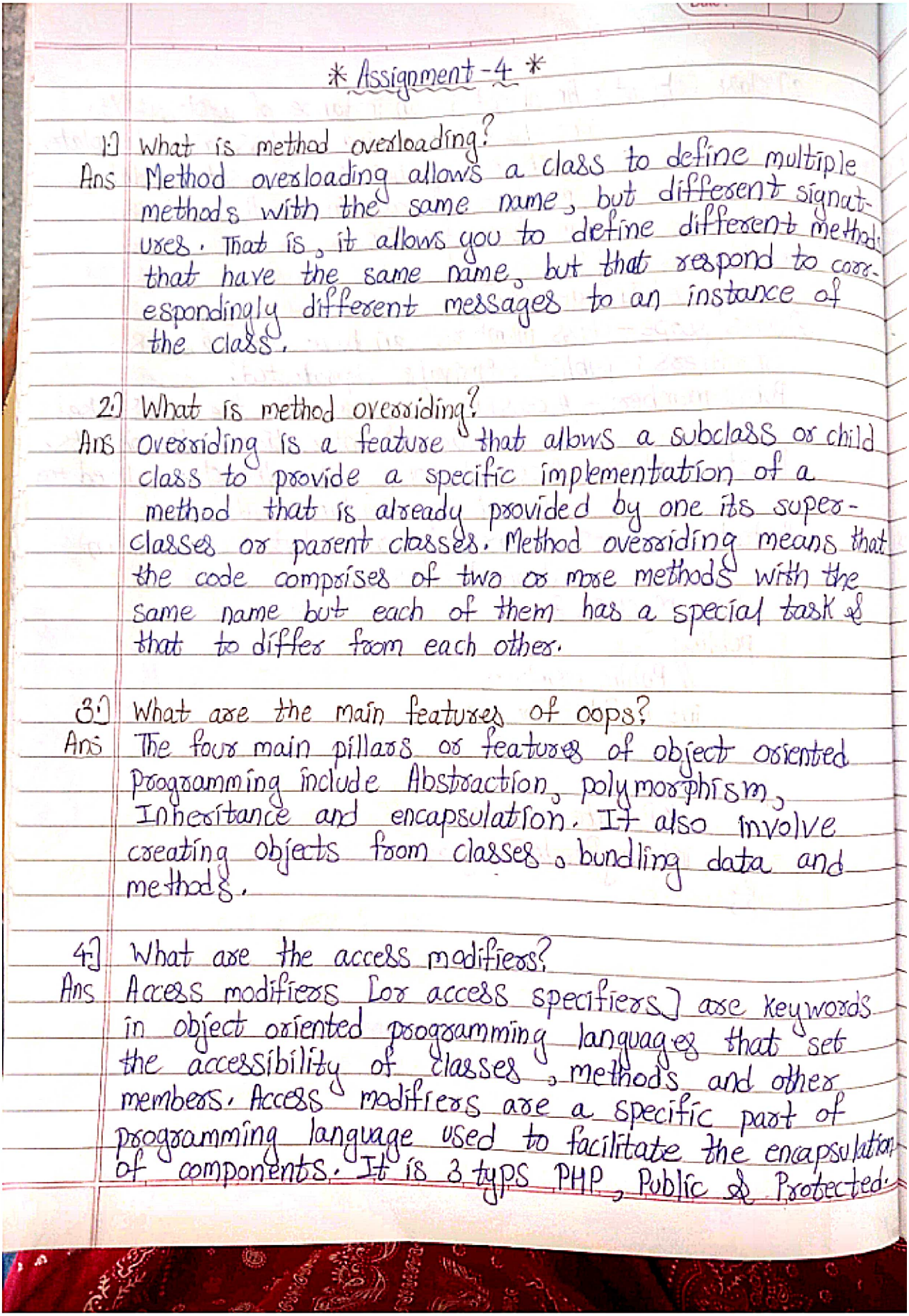
```
class MyClass {
public:
    int myInteger;
    void setInteger(int value)
        myInteger = value;
    }
};
```

2] class object : An object is an instance of a class. You create objects using the class as a template. Each object has its own set of data members & can invoke the class's member functions.

```
Myclass obj1;
obj.setIntegers(42);
```

3] class scope - class members can have diff^n access specifiers : 'public', 'private', 'protected'.
Public members - Accessible from outside the class. Used for interface & interaction with objects.
Private members - Accessible only within the class. Used for internal implementation details.
Protected members - Similar to private, but accessible in derived classes.

```
class MyClass {
public:
    // Public members
    int myPublicInteger;

private:
    // Private members
    int myPrivateInteger;
};
```



* Assignment-4 *

1] What is method overloading?
Ans Method overloading allows a class to define multiple methods with the same name, but different signatures. That is, it allows you to define different methods that have the same name, but that respond to correspondingly different messages to an instance of the class.

2] What is method overriding?
Ans Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one its superclasses or parent classes. Method overriding means that the code comprises of two or more methods with the same name but each of them has a special task & that to differ from each other.

3] What are the main features of oops?
Ans The four main pillars or features of object oriented programming include Abstraction, polymorphism, Inheritance and encapsulation. It also involve creating objects from classes, bundling data and methods.

4] What are the access modifiers?
Ans Access modifiers [or access specifiers] are keywords in object oriented programming languages that set the accessibility of classes, methods and other members. Access modifiers are a specific part of programming language used to facilitate the encapsulation of components. It is 3 typs PHP, Public & Protected.

Page No.
Date :

Q] What is a superclass?
Ans: A class that is derived from another class is called a subclass (also a derived class, extended class or child class). The class from which the subclass is derived is called a superclass (also a base class or a parent class).

Q] What languages come under the oops concept discuss in detail
Ans: OOPs is a programming paradigm that uses objects, which are instances of classes, for organizing code. Several languages embrace the principles of oops. Here are a few =

1] Java - A versatile platform-independent language known as its "Write Once, Run Anywhere" Mantra. Java encapsulates data & behavior within classes & support concepts like inheritance, polymorphism & ensula-encapsulation.

2] C++ - An extension of the c programming language, C++ introduced classes and objects, along with features like inheritance, polymorphism & encapsulation. It allows both procedural and object oriented programming.

3] Python - Although python is a multi paradigm language, it supports object oriented programming. It allows the creation of classes and objects and it include features like inheritance and polymorphism Python's simplicity make it popular for oops beginners.

4] Ruby - A dynamic, reflective and object oriented language, ruby focuses on simplicity and productivity. It supports oop principles like classes, inheritance & encapsulation. Ruby on rails, a web application, framework, is built using ruby.

5] PHP - Originally a server side scripting language. PHP has

Date :

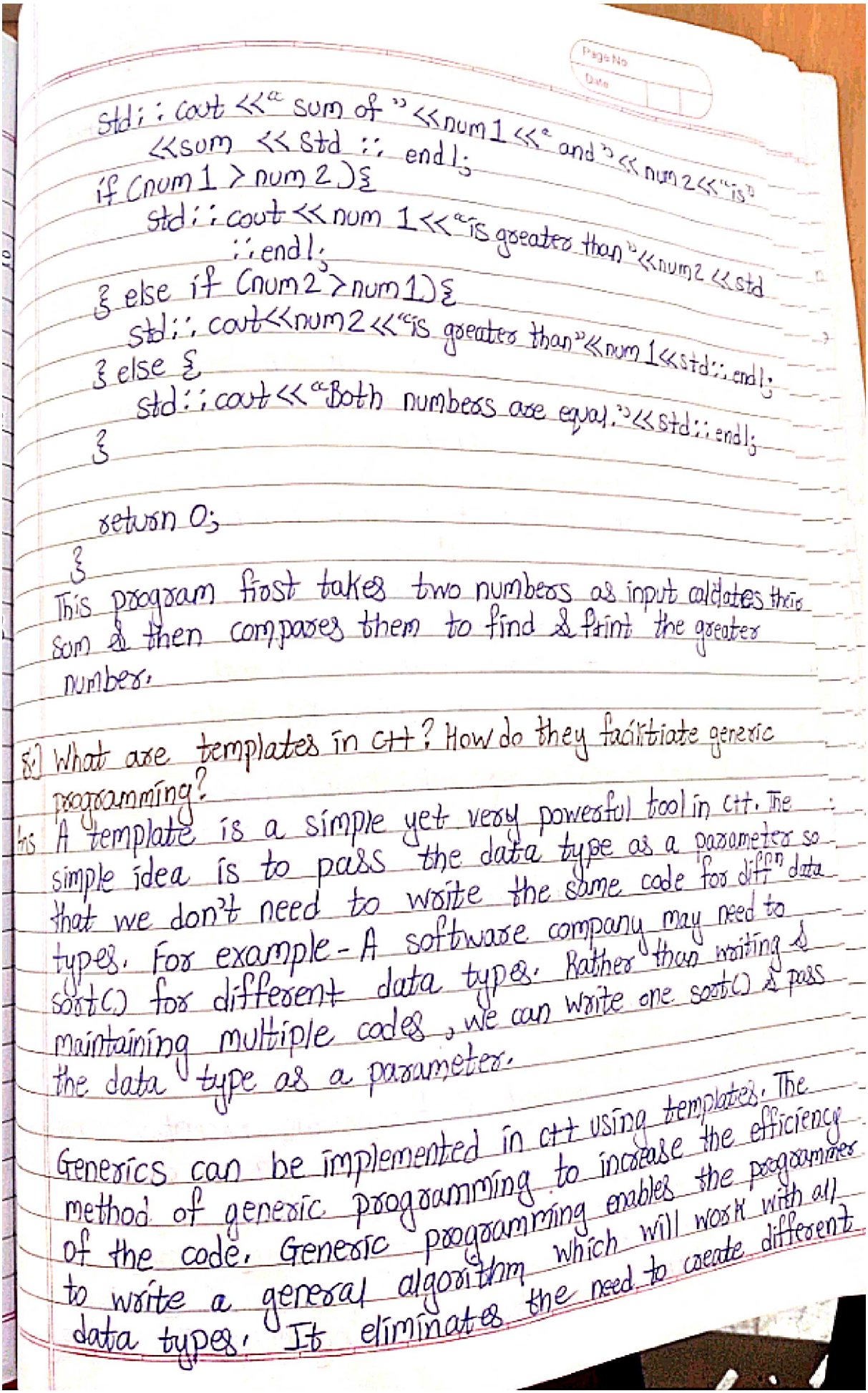
embraced OOP principles with the introduction of PHP. It supports classes, objects, inheritance, polymorphism & encapsulation making it more versatile for larger-scale application.

These languages provide a foundation for implementing OOP concepts, enhancing code organization, reusability & maintainability. Each language has its strengths & use cases, allowing developers to choose the one that best fits their project requirements.

7.) Write c++ program to add 2 numbers & find the greatest of two numbers?

Ans Below is a simple c++ program that takes two numbers as input, adds them and then determines & prints the greater of the two numbers.

```
#include <iostream>

int main() {
    // Declare variables
    double num1, num2;

    // Input: Get two numbers from the user
    std::cout << "Enter the first number: ";
    std::cin >> num1;

    Std::cout << "Enter the second number: ";
    std::cin >> num2;

    // Process: Add the two numbers
    dobble sum = num1 + num2;
```



```
std:: cout << " sum of " << num1 << " and " << num2 << " is "
    << sum << std :: endl;
if (num1 > num2){
    std:: cout << num1 << " is greater than " << num2 << std
        :: endl;
} else if (num2 > num1){
    std:: cout << num2 << " is greater than " << num1 << std:: endl;
} else {
    std:: cout << " Both numbers are equal. " << std:: endl;
}

return 0;
}
```

This program first takes two numbers as input calculates their sum & then compares them to find & print the greater number.

8] What are templates in c++? How do they facilitate generic programming?

Ans A template is a simple yet very powerful tool in c++. The simple idea is to pass the data type as a parameter so that we don't need to write the same code for diff" data types. For example - A software company may need to sort() for different data types. Rather than writing & maintaining multiple codes, we can write one sort() & pass the data type as a parameter.

Generics can be implemented in c++ using templates. The method of generic programming to increase the efficiency of the code. Generic programming enables the programmer to write a general algorithm which will work with all data types. It eliminates the need to create different

Date :

algorithms if the data type is an integers, string or a character.
The advantages of generic programming are-
Code reusability.
Avoid function overloading.
Once written it can be used for multiple time & cases.

We write generic function that can be used for diff^n data types. Example of function templates are sort(), max(), min(), printArray()

9.] Explain the concept of exception handling & discuss its advantages & disadvantages?

Ans Exception handling is a mechanism in c++ that allows you to handle runtime errors or exceptional situations gracefully. It provides a way to separate error-handling code from regular code, improving code readability & maintainability.
In c++, exceptions are raised when errors occur during program execution, & they can be caught & handled by appropriate exception handlers. The key components of exception handling in c++ are:-
1.] Try Block
2.] catch Block
3.] Throw statements.

Advantages - 1.] Separation of concerns- Exception handling separates error handling code from regular code, improving code organization & readability.
2.] Consistent error handling- It provides a consistent way to handle error across diff^n parts of the code.
3.] Robustness- Exception handling helps in creating

robust program by allowing graceful recovery of errors.
4.] Error propagation - Exceptions can be propagated up the call stack until an appropriate handler is found.
5.] Resource cleanup - It allows for proper cleanup of resources.

Disadvantages - 1.) Performance overhead - exception handling may introduce some performance overhead, especially if exceptions are thrown frequently.
2.) Code size - exception handling might increase the size of the compiled code.
3.) Misuse - If used improperly, exception handling can lead to convoluted & hard-to-read code.
4.] Difficult to debug - Debug exceptions can be challenging, especially if not handled correctly.
5.] Compatibility issues - Exception handling may not be compatible with certain resource constrained or real time systems.

12] Difference between structure & classes?

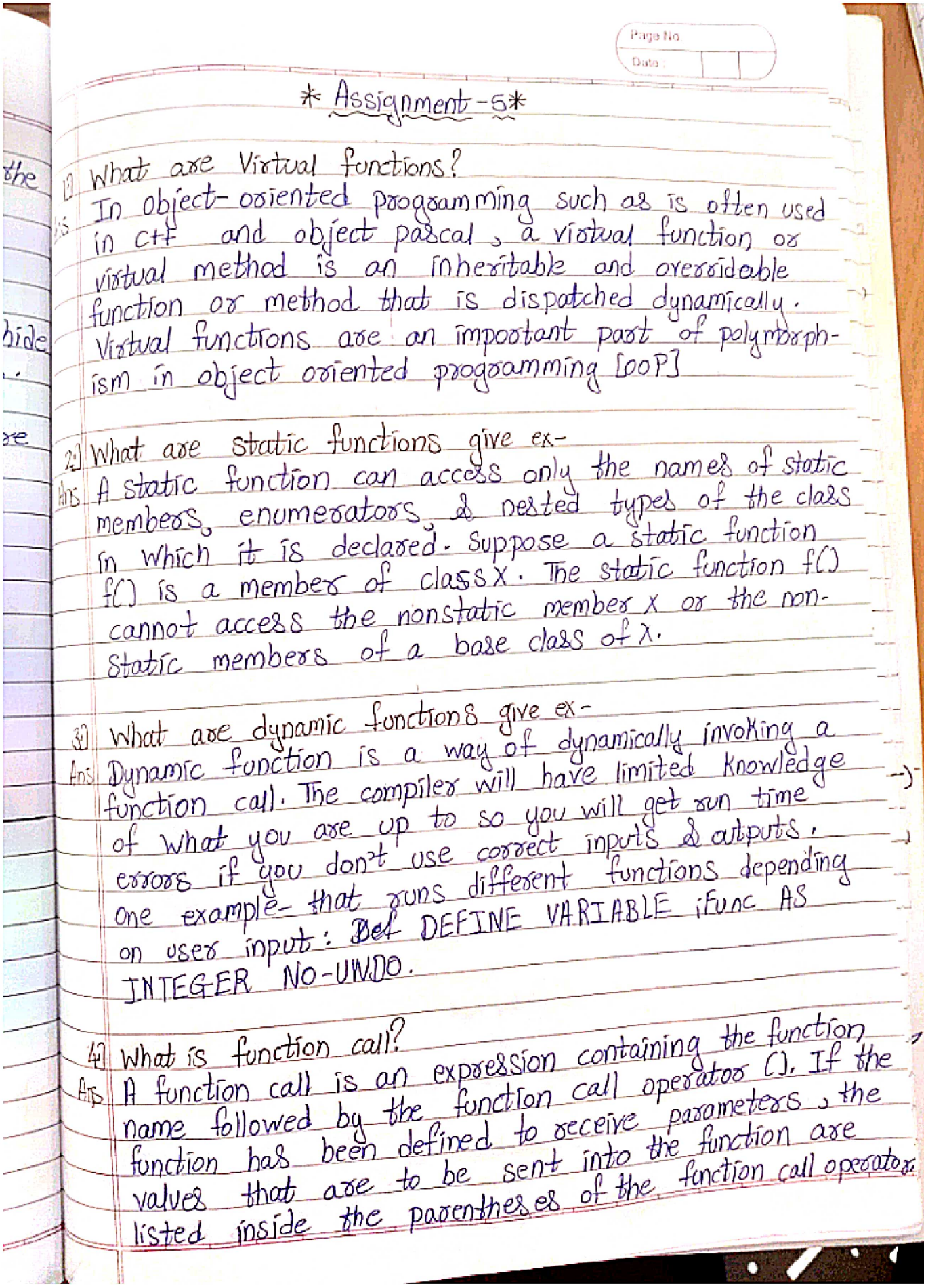
Ans

Class	Structure
1.] Members of a class are private by default	1.] Members of a structure are public by default.
2.] An instance of a class is called an 'object.'	2.] An instance of structure is called the 'structure variable'
3.] Member classes / structures of a class are private by default but not all programming languages have this default behavior eg- Java, etc.	3.] Member classes / structures of a structure are public by default.
4.] It is declared using the class keyword.	4.] It is declared using the struct keyword.

Date :

5.) It is normally used for data abstraction and further inheritance.	5.) It is normally used for the grouping of data.
6.) NULL values are possible in class.	6.) NULL values are not possible.
7.) It is a logical entity. It hides its implementation details.	7.) It will by default not hide its implementation details.
8.) Class requires constructor.	8.) Structure does not require constructure.
9.) It is a reference type	9.) It is a value type.

10.) syntax:
class class_name{
data_member;
member_function;
};
	10.) syntax:
	struct
	structure_name{
	type
	structure_member1;
	type
	structure_member2;
	};



* Assignment-5*

1) What are Virtual functions?

Ans In object-oriented programming such as is often used in c++ and object pascal, a virtual function or virtual method is an inheritable and overridable function or method that is dispatched dynamically. Virtual functions are an important part of polymorphism in object oriented programming [OOP]

2) What are static functions give ex-

Ans A static function can access only the names of static members, enumerators & nested types of the class in which it is declared. Suppose a static function f() is a member of class X. The static function f() cannot access the nonstatic member x or the non-static members of a base class of x.

3) What are dynamic functions give ex-

Ans Dynamic function is a way of dynamically invoking a function call. The compiler will have limited knowledge of what you are up to so you will get run time errors if you don't use correct inputs & outputs. One example- that runs different functions depending on user input: Def DEFINE VARIABLE ifunc AS INTEGER NO-UNDO.

4) What is function call?

Ans A function call is an expression containing the function name followed by the function call operator (). If the function has been defined to receive parameters, the values that are to be sent into the function are listed inside the parentheses of the function call operator.

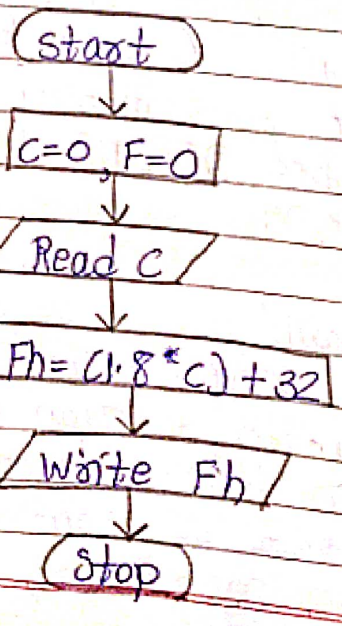
5] What are virtual destructors?

Ans Virtual destructor in C++ in mainly responsible for resolving the problem of memory leaks, when we use a virtual destructor inside the base class it will call the destructor of the child class, ensuring that the child class's object should be deleted & so there might be no memory leakage. If the destruction order of the class objects is incorrect in can lead to what is known as memory leak.

6] Solve the following with algorithm C++ program ex & draw flowchart. Write a program that converts the input converts the input celcius degree into its equivalent fahrenheit degree use the formula;
$F = (9/5)^* C+32$

Ans Algorithm: 1] Start
2] Initialize F=0, C=0
3] Read C
4] Fh = (1.8 * C) +32
5] Print or display Fh
6] Stop

Flowchart:



Page No.
Date :

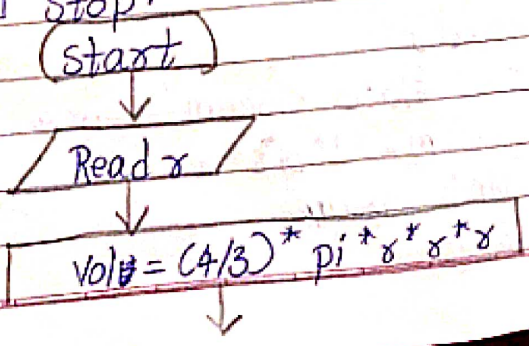
Program:

```
#include<iostream>
using namespace std;

int main()
{
    float C;
    float Fh;

    cout<<"Enter temperature in celsius:"<<endl;
    cin>>C;

    Fh = (1.8 * C) + 32;
    cout<<" Converted Fahrenheit value is:"<<Fh;

    return 0;
}
```

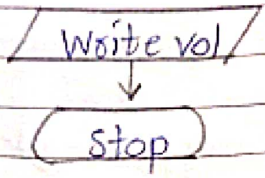
7] Solve the following with algorithm, c++ program example & draw flowchart. Create a program to compute the volume a sphere. Use the formula: $V=(4/3)^* pi^* r^3$ where pi is equal to 3.1416 approximately. The r is the radius of sphere. Display the result.

Ans Algorithm - 1] Start
2] Read r
3] vol = (4/3)* pi * r * r * r
4] print or display volume
5] Stop.

Flowchart -



Date :



Program:

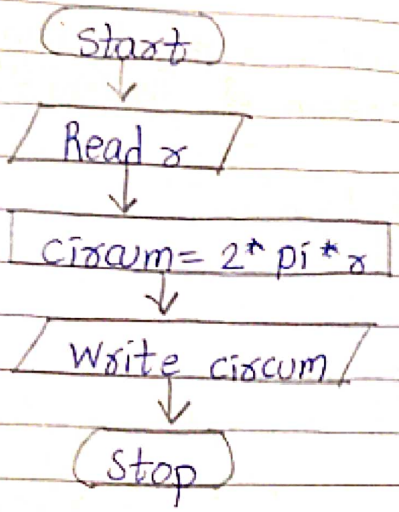
```
#include<iostream>
using namespace std;

#define pi 3.1416

int main()
{
    int r;
    float vol;

    cout<<"Enter radius of sphere: "<<endl;
    cin>>r;

    vol=(4/3)*pi*r*r*r;
    cout<<"Volume of sphere is: "<<vol;

    return 0;
}
```

8.] Design a program to find the circumference of a circle. Use the formula: $C = 2\pi r$, where π is approximately equivalent 3.14.16. With algorithm & flowchart.

Ans Algorithm - 1.] Start
2.] Read r
3.] Calculate circumference by the equation:
Circum = 2 * pi * r
4.] Print circum
5.] Stop

Page No.
Date

Flowchart -



Program -

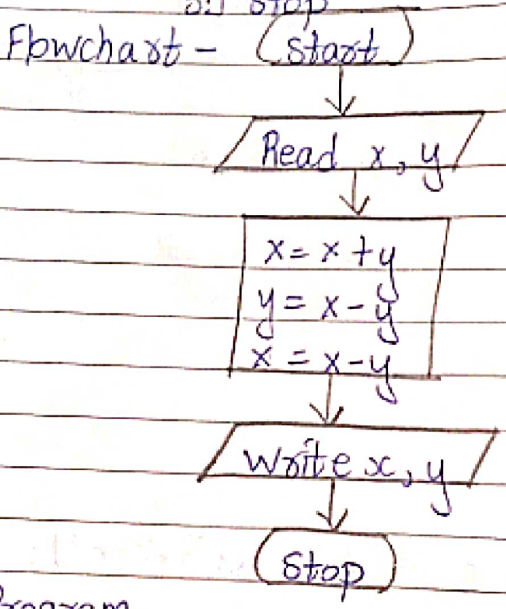
```
#include<iostream>
using namespace std;

#define pi 3.1416

int main()
{
    int r;
    float Circum;
    cout<<"Enter radius of circle r:"<<endl;
    cin>>r;

    Circum = 2*pi*r;
    cout<<"Circle circumference is :"<<circum;
    return
    return 0;
}
```

Date :

Q:] Swapping of 2variables without using temporary or 3rd variable. with algorithm & flowchart.

Ans Algorithm- 1.] Start
2.] Read x & y
3.] x = x + y
y = x - y
x = x - y
4.] Point or display x, y
5.] Stop

Flowchart -



Program-

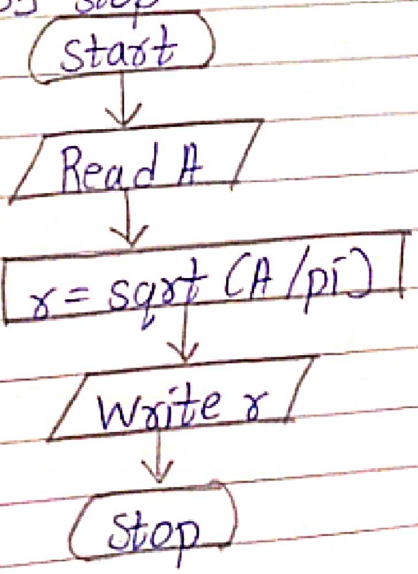
```
#include<iostream>
using namespace std;

int main()
{
    int x, y;
    cout<<"Enter values of x & y:"<<endl;
    cin>>x>>y;

    cout<<"Before swapping, values of x & y are:"<<x<<"
    \t"<<y;
```

```
cout<<endl;

x=x+y;
y=x-y;
x=x-y;

cout<<" After swapping, values of x & y are;"<<x<<
    "\t "<<y;

return 0;
}
```

10] Write the program to compute the radius of a circle. Derive your formula from the given equation: $A = \pi r^2$, then display the output.

Ans Algorithm - 1] Start
2] Read r
3] Calculate radius by the equation:
$r = \text{sqrt}(A/pi)$
4] write r
5] stop

Flowchart -



Date :

Program:

```
#include<iostream>
#include<math.h>
#define pi 3.1416
using namespace std;

int main()
{
    int A;
    float r;
    cout<<"Enter area of circle A:"<<endl;
    cin>>A;

    r= sqrt(A/pi);
    cout<<"Radius of circle is:"<<r;

    return 0;
}
```