Principles Of Object-Oriented Programming

First OOPs language → Simula 1967

C++ is a high level, general-purpose programming language created by Danish computer scientist Bjarne Stroustrup in 1985.

Family : C

It is a combination of Simula and C.

Paradigms : Multi-Paradigm : Procedural, imperative, functional, object-oriented, generic, modular.

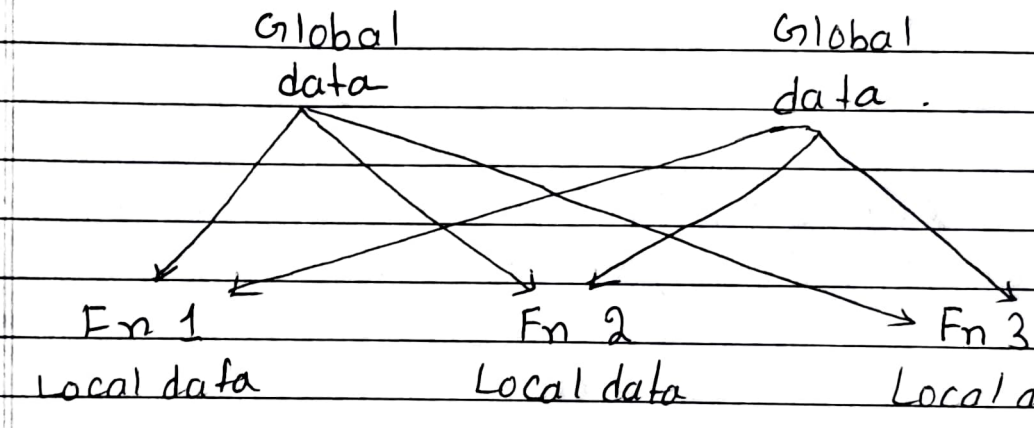
Stable release version : C++20

Procedure-Oriented Programming (POP)

In POP, the problem is viewed as sequence of things to be done such as reading, calculating and printing. A number of functions are written to accomplish these tasks. POP consists of writing a list of instructions for the computer to follow and organising these instructions into groups known as functions.

In POP, Global data are more vulnerable to an inadvertent change by a function. In a large problem it is very difficult to identify what data is used by which function.

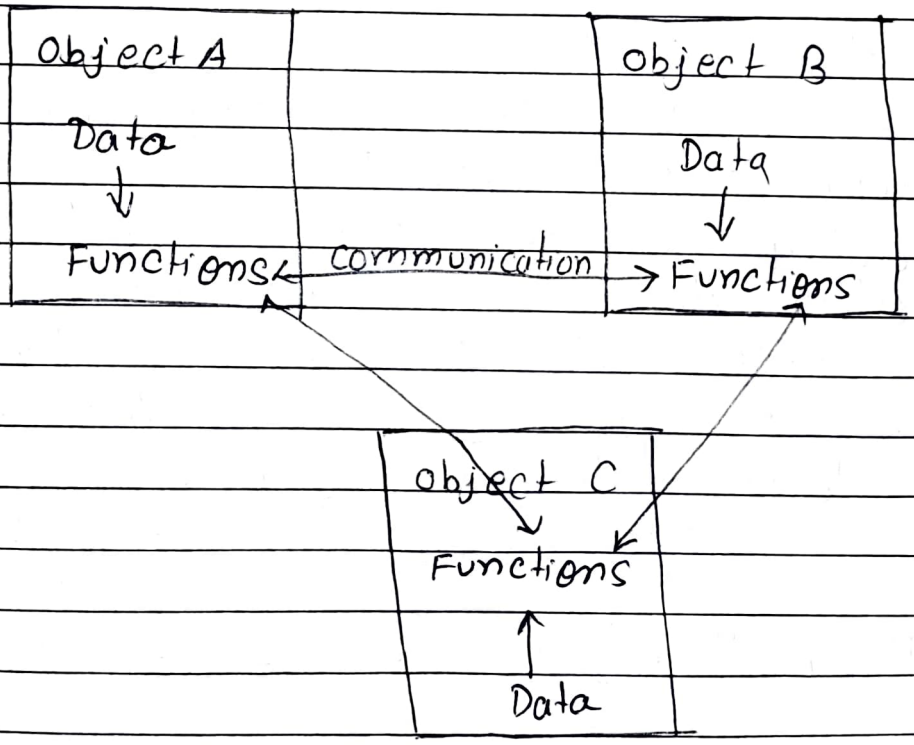
It employs top-down approach in program design.



Object-Oriented Programming Paradigm

OOPs treats data as a critical element in the program development. It protects data from accidental modifications from outside functions. It allows decomposition of problem into a number of entities called objects.

- Emphasis is on data.
- Programs are divided into what are known as objects.
- Data is hidden and cannot be accessed by external functions.
- Object may communicate with each other through functions.
- Follows bottom-up approach in program design
- New data and function can be easily added whenever necessary.



OOPs Terminology

1) Objects.

Objects are the basic run-time entities in object-oriented system. They can interact without having to know details of each other's data or code.

2) Classes.

The entire set of data and code of an object can be made a user-defined data-type with the help of a class. A class is a collection of object of similar type. Objects are variables of type class. We can create any number of objects belonging to that class.

3) Data Abstraction and Encapsulation.

The wrapping up of data and function into a single unit (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.

The insulation of data from direct access by the program is called data hiding.

Abstraction refers to act of representing essential features without including the background details or explanations. Classes use the concept of abstraction and are defined as a list of abstract attributes and functions to operate on these attributes.

The attributes are called data members because they hold information.

The functions that operate on these data are called methods ~~of~~ or member functions.

Since, the classes use the concept of data abstraction, they are called as Abstract Data Types.

4) Inheritance.

Inheritance is the process by which objects of one class acquire the properties of objects of another class. It provides the idea of reusability. This means that one can add features to an existing class without modifying it.

5) Polymorphism.

An operation may exhibit different behaviour.

The process of making an operator to exhibit different behaviour in different instance is known as operator overloading. for Eig;

```
int a = 5, b = 6;
int c = a + b;          // output : 11.

string a = "S", b = "N";
string c = a + b;       // output : SN
```

Using a single function name to perform different tasks is known as function overloading.

for e.g;

```
int sum(int a, int b);
float sum(int a, float b);
int sum(float a, float b, int c);
```

Function overriding is concept when we define a function of same name and same function signature (parameter and data type) in both the base and derived class with a different function definition.

```
int sum(int a, int b)
{
    return a+b;
}

int sum(int a, int b)
{
    return a+b-1;
}
```

21/10/2023

Ajanta
Page No.
Date

6) Dynamic Binding

It refers that the code associated with a given procedure is not known until the time of the call at run-time.

7) Message Passing

It involves specifying the name of object, function (message) and the information to be sent.

8) Application of OOP

Real time business systems are often much more complex and contain many more objects with complicated attributes and methods.
The Promising areas for application of OOP include:

- Real-time systems
- AI and expert system.
- Neural networks and parallel Programming.
- CIM/CAM/CAD systems.

22105128040

Ajanta
Page No.
Date

object-based Languages

- Data encapsulation
- Data hiding and access mechanism
- operator overloading
- Automatic initialization & Clear-up of objects.

Object-oriented Languages.

object-based features + inheritance + dynamic binding.

21/10/2023.

Ajanta
Page No.
Date

22105128040

Beginning with C++.

What is C++?

- C++ is a OOP language.
- Developed by Bjarne Stroustrup at AT&T Bell Laboratories in Murray Hill, New Jersey, USA.
- Superset of C
- Extension of C with a major addition of class construct feature of Simula 67.

Applications of C++.

C++ is a versatile language for handling very large programs.

- It allows us to create hierarchy-related objects.
- It is able to map real-world problem properly, the C Part of C++ gives the language the ability to get close to the machine-level details.
- C++ programs are easily mainable and expandable.

Simple C++ Program features

Every C++ Program must have a main().
With a few exceptions, the compiler ignores carriage returns and white spaces.
Like C, the C++ statements terminate with semicolon (;).

Comments

```
// single-line comment

/* Multi
line
comment */
```

~~Output Obje.~~

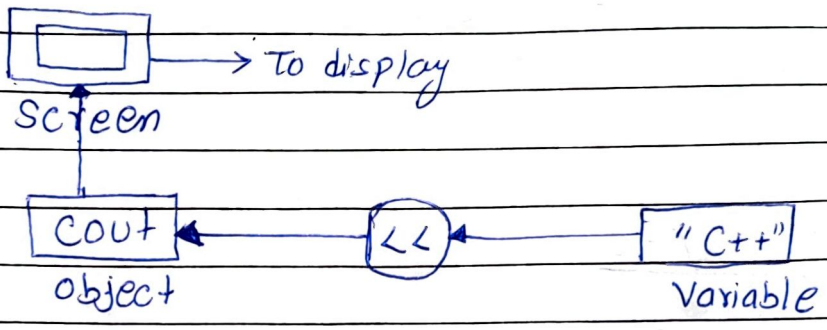
Output Operator.

```
cout<<"C++ is Simple";
```

The above statement introduces two new C++ features, cout and <<.
cout → 'C out' is a predefined object that represents the standard output stream in C++.

Contd:

The operator << is called the put to or insertion operator. It sends the contents of the variable on its right to the object on its left.



output using insertion operator.

The iostream File

```
#include <iostream>
```

This directive causes the Preprocessor to add the contents of the iostream file to the program. Its holds the declarations for cout, and the operator <<.
It contains function Prototypes for the standard input and output functions.

Header file.

New-version

<ctype.h> → contains function prototypes that can be used to convert lowercase to upper case and vice-versa. ← <cctype>.

<float.h> → contains the floating-point size limits of the ~~function~~ System. ← <cfloat>.

<math.h> → contains function prototype for math functions ← <cmath>.

<stdlib.h> → contains function prototype for conversion of number to text, memory allocation and other utility functions. ← <cstdlib>

<string.h> → contains prototypes for C-style string processing functions ← <cstring>

<stdio.h> → contains function prototype for standard input-output library functions. ← <cstdio>

<fstream.h> → contains prototypes for < <fstream> functions that perform input from files on disk and output to files on disk.

Namespace

It is a new concept introduced by ANSI C++ Standards. This defines scope for the identifiers that are used in a program.

For using the identifiers defined in the namespace scope –

```
using namespace std;
```

std is the namespace where ANSI C++ Standard class libraries are defined.

using, namespace are new keywords of C++.

Return type of main().

Every main() in C++ should end with a return 0; statement; otherwise a warning or an error might occur.

Since main() returns an integer type value, return type for main() is explicitly specified as int.

By default return type for all functions in C++ is int.

```
main()
{
    ...
    ...          ⟶ run with a warning.
}
```

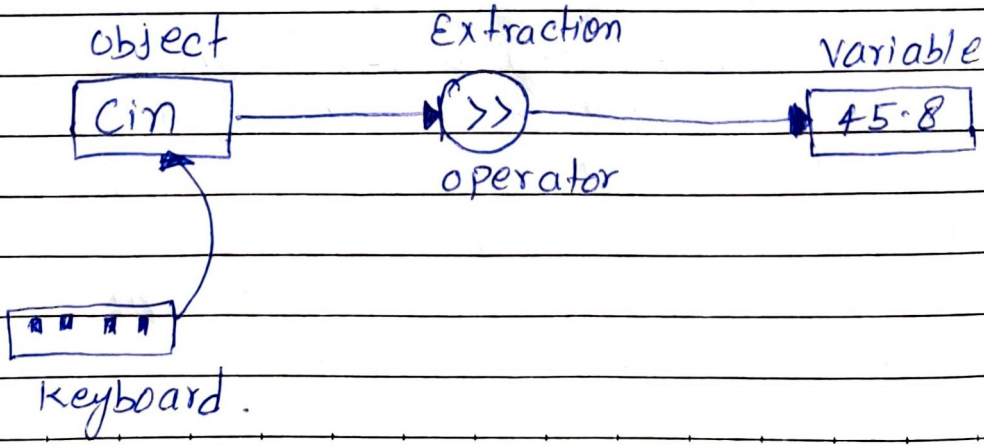
Input Operator

cin >> num; → 'C in'
is an input statement and causes the program to wait for the user to type in a input from keyboard. cin is a predefined object in C++ that corresponds to the standard input stream. Here, this stream represents the keyboard.

The operator >> is known as extraction or get from operator. It takes the value from the keyboard and assigns it to the variable on its right.

Like <<, >> can be used as bitwise left or right shift. This is the example how one operator can be used for different purposes. This concept is known as operator overloading – an aspect of polymorphism.

It is important to note that we can still use scanf() and printf() functions. C++ accepts this notation. To maintain the spirit of C++, we will use cout, cin.



21/10/2023.

Page No.
Date

Cascading of I/O operators

```
cout<<"Sum =";
cout<<sum;
cout<<"\n";
```

OR.

```
cout<<"Sum ="<<sum<<"\n";
```

output:
Sum = 40 (in both cases)

Same lies for cin and >> also.
This saves the lines and makes code redundant.

//2210512804O.

Structure of C++ Program.

- Include files.
- Class declaration
- Member function definitions.
- Main function Program.

Creating Source file

Syntax: <filename>.cpp

2210512804O 22/10/2023

Tokens, Expressions and Control Structures

Tokens - It is the smallest individual units in a program are known as tokens. C++ has the following tokens.

- Keywords
- identifiers
- constants
- strings
- operators.

Keywords-
They are explicitly reserved identifiers and cannot be used as names for the program variables or other user-defined Program elements.

~~ANSI C++ places no limit on its length and~~
There are 63 keywords in C++.
Keywords added by ANSI C++

bool export reinterpret_cast typename
const_cast false static_cast using
dynamic_cast mutable true wchar_t
explicit namespace typeid asm
catch class delete friend inline
new operator private protected public

template this throw try virtual.

Wchar_t type is a wide-character literal introduced by ANSI C++ and is intended for character sets that cannot fit a character into a single byte. Wide-character literals begins with L. for e.g, L'ab'

Identifiers and Constants

Identifiers refers to the name of variable, functions, arrays, classes, etc created by programmer.

abl ✓ 1ab X -1ab ✓

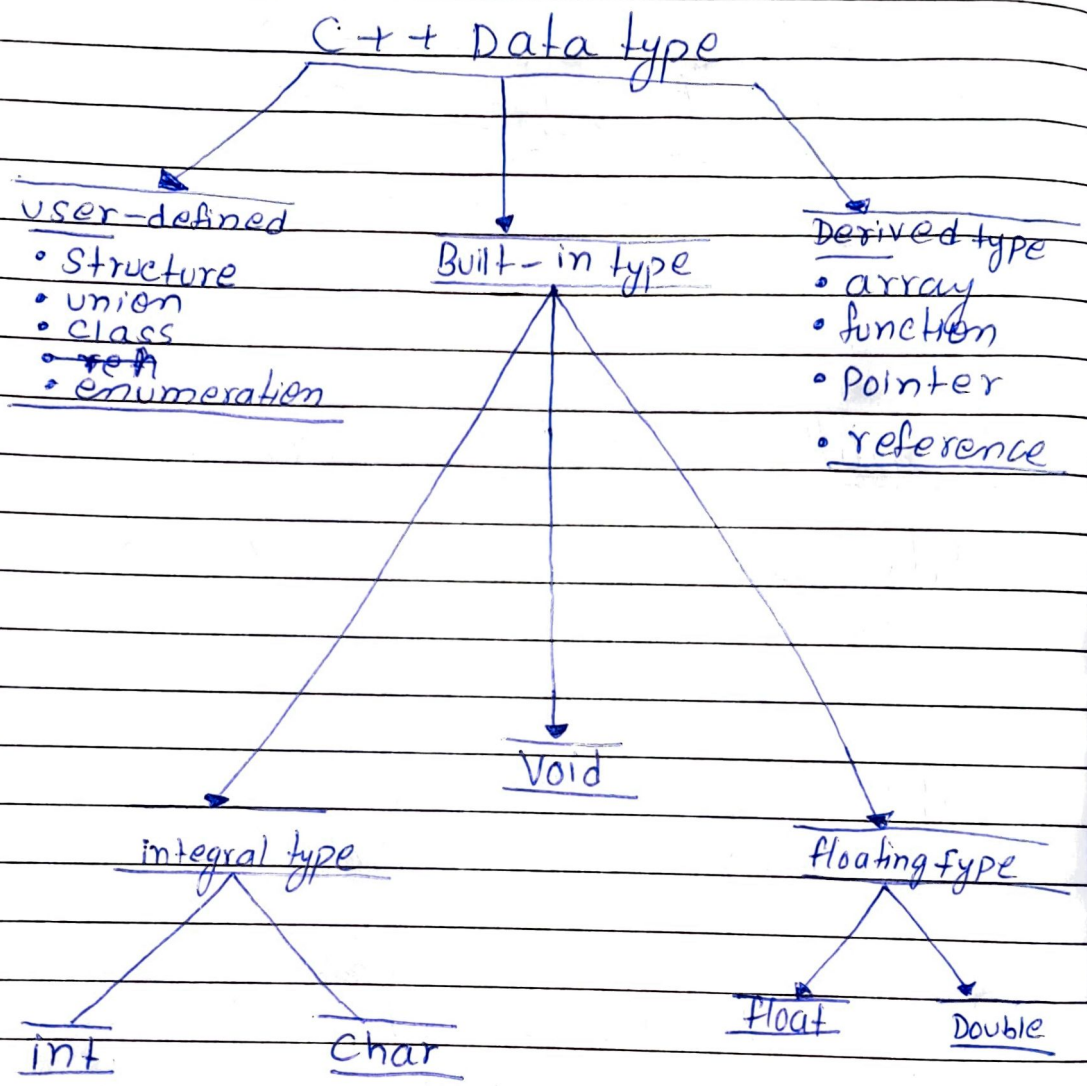
<keyword> X a ≠ A

Constant refers to ~~fixed~~ fixed values that do not change during the execution of a program. Const creates typed constant.

Variable refers to entity whose value may change during the execution of a program.

#define create constants that have no type information.

Basic Data Types



VOID Type

The two normal uses of void are -

1) ① to specify the return type of a function when it is not returning any value.
② to indicate an empty argument list to a function.

E.g; void fun(void);

2) In the declaration of generic pointer. Generic pointer can be assigned a pointer value of any basic data type.

```
void *gptr;        or   int *ptr;
                        gptr = ip; //int to void ptr
```

A void pointer cannot be directly assigned to other type pointers in C++. We need to use a cast operator -

```
ptr2 = (char *)ptr1;
```

User-Defined Data Type

Structure (struct and union) are same as C in C++.

In C++, Class. are just like any other basic datatype, to declare variables. The class variables are known as objects.

Enumerated Data type

enum in C++, the tag names becomes new type names. By using these tag names, we can declare new variables.

Eg; C++ does not permit an int value to be automatically converted to an enum value.
An enumerated value can be used

In C++, an enum defined within a class (or structure) is local to that class (or structure) only.

```
enum shape{circle, square};

shape ellipse;  //ellipse is of type shape.

shape ellipse = circle;   //allowed.
shape ellipse = 7;        //Error.
shape ellipse = (shape)7; //ok.

enum color{ red, blue = 5, green = 9};

enum { off, on};  -> C++ allows creation of anonymous enums (without tag name).
```

Derived Data type

1. Arrays –
They are the collection of similar data type. In C++, the size of array be one larger than the character in string.

```
char string[3] = "abc"; // valid in c but
                        not in C++.

char string[4] = "abc"; // vaid in C++ also.
```

2. Function

3. Pointer.

C++ adds the concept of Constant pointer and pointer to a constant.

```
char * const ptr1 = "GOOD"; // cons. ptr.

int const * ptr2 = &m; // pointer to a constant.
```

Pointers are extensively used in C++ for memory management and achieving polymorphism.

Type casting

C++ Permits explicit type conversion of variables or expression using the type cast operator.

```
type_name (expression)   // C++ notation
```

E.g.:

```
avg = sum / float(i);
```

A type-name behaves as if it is a function for converting values to a designated type.

New cast operators –

- const_cast
- static_cast
- dynamic_cast
- reinterpret_cast.

Operators in C++

:: - Scope resolution operator.
::* - pointer-to-member declarator.
->* - pointer-to-member operator.
.* - pointer-to-member operator.
delete - Memory release operator.
endl - Line feed operator (in output statement)
new - Memory allocation operator
setw - Field width operator. → right justified.

Scope-Resolution Operator.

In C, the global version of a variable cannot be accessed from within the inner block. C++ resolves this problem by introducing a new operator :: called scope resolution operator. This is used to uncover a hidden variable.

:: var_name.

<iomanip> → endl, setw.

Expressions and their types

A expression is a combination of operators, constants and variables arranged as per the rule of language.

- constant exp.
- Integral Exp.
- Float exp.
- Pointer exp.
- Relational Exp.
- Logical exp.
- ~~Logic~~
- Bitwise exp.

An expression made up of combination of above expressions called as Compound expressions.

	Operator	Associativity.
1.	:: (postfix)	L-R.
2	-> . () [] ++ --	L-R
3	prefix(++ --) ~ ! unary + unary - unary & (type) sizeof new delete.	R-L
4	-> * *	L-R
5.	* / %	L-R

22105128040

22/10/2023.

6.	+ -	L-R.
7	<< >>	L-R
8.	<< = >> =	L-R
~~9.~~	~~= = !~~	
9	= = !=	L-R.
10.	&	L-R
11	^	L-R
12	|	L-R
13	&&	L-R
14	||	L-R
15	? :	L-R
16.	= * = / = % = + = =	R-L.
17	<< = >> = & = ^ = | = ,	L-R.

The unary assume higher precedance.

L-R → Left to right
R-L → Right to Left.

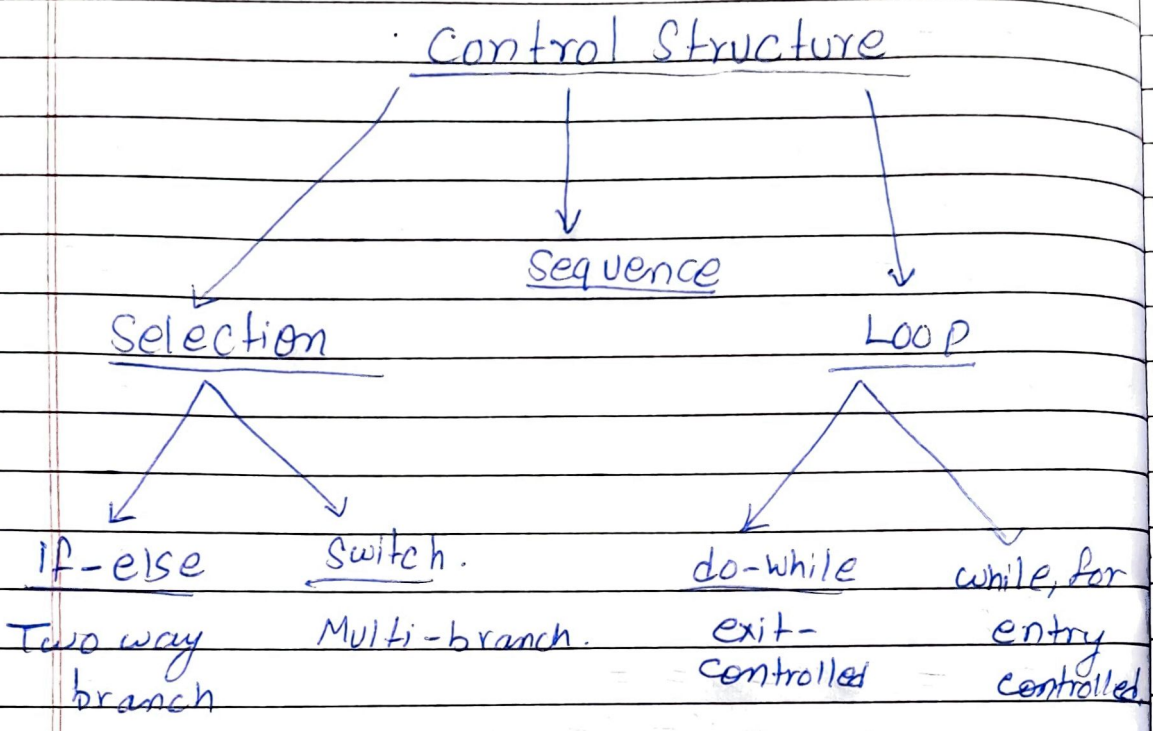
Reference variable

datatype & reference-name = var-name;

↓

Syntax

22/10/2023.



1). do
{
action1;
}
while (condition);

2) while (condition)
{
action;
}

3) for (initialization; condition; increment)
{
action;
}

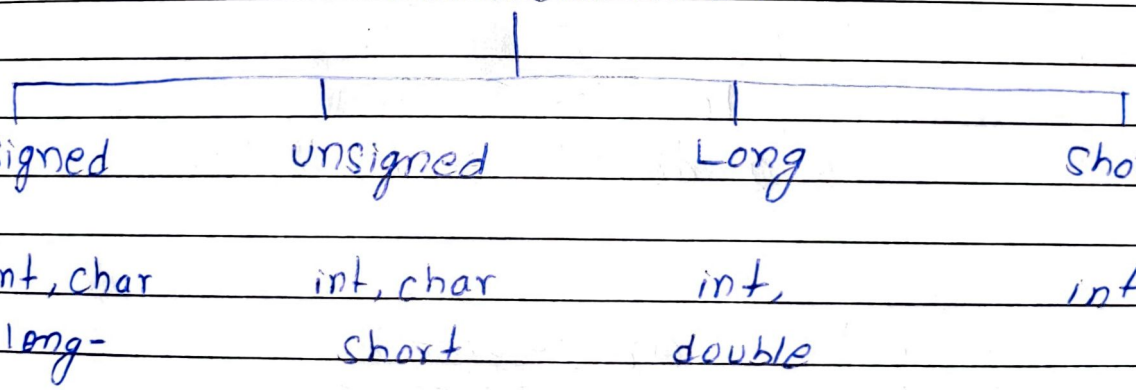
28/10/23.

Page No.
Date

Overloaded Constructor

Type Modifiers

Type Modifiers are used in C++ to give extra meaning to already existing data types. It's added to primitive data types as a prefix to change the meaning of a basic data type.



Signed Modifiers –
It Stores positive, negative integer ~~to~~ or zero value in a signed variable.

Unsigned Modifier –
Stores only non-negative integer values.

short –
Modifies the minimum value that a data type can hold.

long –
Modifies to the maximum value that a data type can ~~long~~ hold.

signed/unsigned short → 2 byte.
signed/unsigned long → 8 byte.
float → 4 byte.
double → 8 byte.
long long → 8 byte.
long double → 16 byte.

Symbolic Constants

There are two ways of creating

1). using the qualifier 'const'.
2) Defining a set of integer constant using 'enum' keyword.

i). In C++, we can use const in a constant expression, as –

const data-type var-name = value;

```
const int a = 10;
char name[a];
```

const allows us to create typed constants instead of having to use #define to create constants that have no type information

With long and short, we can use -

```
const a = 10;   ≈ const int a = 10;
```

it defaults to int.

C++ requires a const to be intialized, if none; it initializes to 0.

To give a const value an external linkage so that it can be referenced from another file, we use extern in C++.

```
extern const size = 10;
```

In ANSI C or C, const are global but in C++ it is local. In C, we are using 'static' to make const to behave in local scope.

2) enum { x, y, z };

enum { x = 40, y = 50 };

Reference Variable

It provides an alias (alt- name) for a previously defined variable.

data-type & ref-name = var-name;

float total = 100;
float & sum = total;
↳ initialization.

Sum is the alternative name declared to represent the variable total.

cout << total; & cout << sum;
will print 100.

total = total + 10; → 110 (total, sum)

sum = 0 → total = 0 / sum = 0.
↳ assignment

initialization ≠ assignment.

A reference variable must be initialized at the time of declaration. This establishes the correspondence between the reference and data object.

Strings

C++ strings are sequence of characters stored in a char array. Strings are used to store words and text. They are also used to store data, such as numbers and other types of information.

C style → char str = "suhani";

C++ style → string str = "suhani";

Reading a string –

1) using cin

```
cin >> str;
```

2) using getline.
→ used to read a string from an input stream. It has declaration in <cstring> or <string.h>

```
getline(cin, str);
```

3) using g. stringstream
→ used to take multiple string as input at once

String Passing to functions

```
void print-string ( string str)
{
    cout << str << endl;
    return;
}
```

Pointer and Strings

```
string str = "Subani";
char *ptr = &Str[0];
while (*ptr != '\0')
{
    cout << *ptr;
    ptr++;
}
cout << endl;
return 0;
```

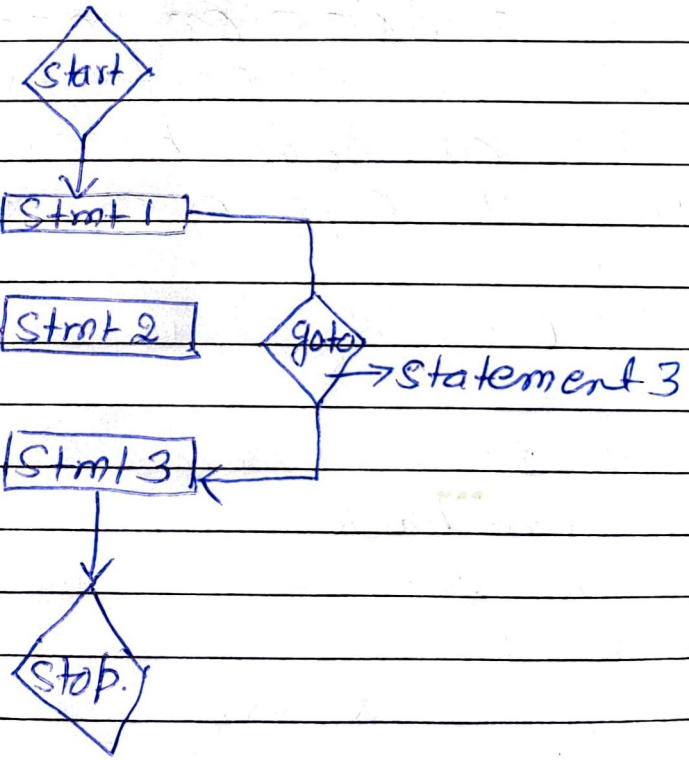
String functions

length(), swap(), size(), clear()
strncmp(), strncpy(), strrchr(),
strcat(), compare()

goto statement

It provides an unconditional jump from the goto to a labeled statement in the same function.

use of goto is discouraged as it makes difficult to trace the control flow of a program.



Break and Continue

Break stops the entire process of the loop. Continue statement only stops the current iteration.

Switch case

It is a decision making statement that can execute the different blocks of code based on the value of ~~expression~~. expression.

```
switch (condition)
{
    case 1:
        code block.
        break;
    case 2:
        - - - .
        - - .
    default:
        code block
        break;

}
```

Conditional operator

expression ? statement 1 : statement 2;

if expression / condition is true, statement 1 will be executed and vice versa.

Memory Management Operators

C uses malloc() and calloc() functions to allocate memory dynamically at run time.

In C++, it also defines two unary operators 'new' and 'delete' that perform the task of allocating and freeing the memory in a better and easier way. They are free-store operators as they manipulate memory on the free store.

An object can be created by using new and destroyed by using delete.

pointer-variable = new data-type.

- → is a pointer of type data-type
- → holds the address of memory space allocated.

```
p = new int;
q = new float;
```

OR

```
int *p = new int;
float *q = new float;

*p = 25;
```

```
pointer-variable = new data-type(value);
```

↳ initialization of memory.

```
int *p = new int(25)
arr-ptr = new int[3][3];
```

When a data object in no longer needed, it is destroyed to release the memory space for reuse.

```
delete pointer-variable;

delete p;
```

if we want to free a dynamically allocated array,

28/10/23

delete [size] pointer variable;

delete [] size; delete [] p;
↳ delete entire array.

Size refers to the number of elements in a array to be freed.

Advantage of new over malloc().

1) new and delete can also be overloaded

2) It is possible to initialize the object while creating the memory space.

3) It automatically computes the size of the data object.

4) It automatically returns the correct pointer type.

Functions in C++

```
void fun_name();  // function declaration.

void func_name()
{
    ----.
    ----.
}                 // function definition.
```

When the function is called, control is transferred to the first statement in the function body.

In C++, the main() function returns a value of type int to the operating system.

```
int main();
int main(int argc, char *argv[]);
```

Function having a return value should use the return statement for termination.

Function Prototyping

In function prototyping with C++, a template is always used when declaring or defining a function. When a function is called, the compiler uses the template to ensure the proper arguments are passed and the return value is treated correctly.

Function prototype is a declaration statement.

type func-name (argu-ment list);

Eg; int volume(int x, int y, int z);

for a void argument in C++,
void display(void);

Call by Reference

Provision of reference variables in C++ permits us to pass parameters to the function by reference. When we pass arguments by reference, the formal argument in called ~~calling~~ function becomes the aliases to the 'actual' argument in calling function. This means that when the function is called working with its own arguments, it is actually working on original data.

```
Void swap(int &a, int &b)
{
    int z = a;
    a = b;
    b = z;
}
```

we can call it —
swap(m, n).

Return by Reference.

A function can also return a reference.

```
int &max(int &x, int &y)
{
    if (x > y)
        return x;
    else
        return y;
}
```

The function will return reference to x or y, not the values.

Inline function

A function that is expanded in line when it is invoked. The compiler replaces the function call with corresponding function code.

```
inline function-header
{
    function body
}.
```

The speed benefits of inline may diminish as the function grows in size

Default Argument

Default values are specified when the function is declared.

```
float SI(float p, int t, float r=5);

value = SI(500, 7);
```

function will use default value.

Function Overloading

→ refers to the use of same function for different purposes. It uses same function name to create function that perform a variety of different tasks. This is known as polymorphism in OOP.

```
// Declarations:
1. int add(int a, int b);
2. int add(int a, int b, int c);
3. double add(double x, double y);
4) double add(int p, double q);

// function calls.

add(5, 6);        // uses 1
add(5.6, 9.7);    // uses 3
add(5, 9.7);      // uses 4.
add(50, 40, 40);  // uses 2.
```

Steps -

→ Compiler finds an exact match in which type of actual argument are same and uses that function.

28/10/23

2) if match is not found, the compiler uses integral promotions to actual arguments.

char to int
float to double
to find a match.