All about C++ in Depth (theory, practical)

- What is program – set of instructions in lang. for (comp. to execute.
- What is C++? Explain

→ C++ is a high level Semi-object oriented programming language (we can study without learning OOPS) developed by "Bjarne Stroustrup" in yr. 1979 in Bell lab

Note – (i) C++ is a case sensitive language (A or a matters)

(ii) Earlier name of C++ was :- C with Classes.

(iii) C++ is a portable programming language. (like win 7 to win 7 only, same bit computer) operating

Syntax of C++ Programs.

e processor
tive

```
#include <iostream.h>  -(Compulory)   ] Header files.
#include <conio.h>  (optional)

Eg - int, char, void, double
return type  main()
{
    // code ;

}
```

- Starting and ending point is main function [main()]

For input, use → cin >>
For output, use → cout<<] (included in "iostream" only)

A header file → #include <iomanip>

If done → cout << fixed << set precision (4) << sum;

then op be printed till 4 decimal places.

Compilation & Execution process of C++ program.

First.cpp (We saved as)

→ Preprocessor (links header files) (#...)

→ compiler

→ first.obj file

→ Linker (add more than one object file & libraries)

→ first.exe (loads .exe file into main memory)

→ Loader

→ Program runs

Q What is data type?

→ Data type defines the type/kind of value the variable will store. eg - int a.

C++ has 3 types of datatype

(i) Basic

(ii) Derived

(iii) User-defined.

Basic Datatype :-

	Size	Range
• int	4 bytes	-32768 to 32768
• char	1 byte	-128 to 127
• float	4 byte	-3.4×10^{-38} to $3.4 \times 10^{+38}$
short int	2 bytes	
• double	8 bytes	
• bool	1 byte	True or false.
• void	no size	

Derived Data type

- Array
- Function
- Pointer
- Reference.

User-defined

- Structure
- Union
- Class
- Enumeration.

~~Signed~~ — ~~Both +ve an~~
~~unsigned~~ — ~~Only +ve in~~

Eg - signed int a = 10
or
-10 b

but
unsigned int b = -20
not

What is overflow ?

→ Eg - ~~long~~ int a = 1212121
int b = a ;

(~~Storing longer data to sm~~
~~datatype due to insuff~~
~~some memory get l~~

DATE: / /

@ What is Variable?

→ Variable is the name of memory location where we store value.

- Variables are case-sensitive in C++.
(We can also change the value as variable's name tells.)

- In C++, variable must start with either (a-z, A-Z) or _ (underscore)

- We can't give extra space btwn variable.

ab = 10 ✓

a b = 10 ✗
(space)

Variables are of 3 types :-

```
# include <iostream>
using namespace std;
int b = 20 ; // global variable ✓
static a = 10; // static ✓

void main() {
    int d = 70; // local ✓
    static c = 80; // static
    cout ~
```

How char ! related to it ?
~~int Kelly = 'A'~~

Q What is Key word?

→ keyword is a reserved word, whose meaning is already defined on the compiler. Eg, void, int, or anything that comp. knows

- We can't use keyword as a variable & constant name. Eg - int float=10 X
- Keywords must be in lower case.

For eg → int, char, float, static, inline, template, class, struct etc.

→ Total keyword - 48 + ...

→ Also Matlab jo chiz comp. jo already chiz jese int, const, etc jaanta hai usko as a variable use nhi kar sakte. But INT or Int can be used rather than int. Iska color change ho jaye, wo keyword hai.

Q What is Constant?

→ It is a fixed value which can't be changed in a run time.

Note -
- Any datatype be of
- Also called literals.
- But using pointer we can change value of constant.

Eg - const int a = 10;
~~int~~ a = 20;
cout<<a; → gives error then

Q What is Identifier?

→ Name used to Identify a variable, function, class, module or any other user-defined items is called identifier.

→ keyword not allowed.

Eg – int abc = 10
↳ Identifier.

Rules →
(1) Contain letters/upper digit, _.
(2) nondigit char
(3) No special char like @.

Q What is operator?

→ Symbol used to perform mathematical and logical task is k/a operator.

C++ operators →

(1) ~~Arithmatic~~ Relations operator (>, <, =, >=, ==, !=)
(2) Logical operator (&&, ||, !)
(3) Assignment operator (=, +=, -=, *=, /=)
(4) Arithmatic operators (+, -, *, /, %) → Used as char = *
(5) Ternary " (? :) → Like if else → weather = (temp>30) ? "hot" : "
(6) Bitwise " (&, |, ^, ~, <<, >>)

(Modulo means %)

→ Conditions → if, if-else, nested if-else, else if ladder, switch statement.

Not imp

if → if (cond.) { cout << } → if temp > 30 print "hot" otherwise cold

if-else → if (cond.) { cout } else { statement }

→ ~~Nested~~ if-else ladder → ~~then~~

Syntax → if (cond.) { statement 1 } ;
else if (cond.) { statement 2 } ; ...
else { statement 3 } ;

→ else - if ladder →

```
→ if (cond) {
        if (cond.) {
            Statement ;
        }
        else
        {
            statement ;
        }
    }
else
{
    if (cond.)
    {
        Statement ;
    }
    else
    {
        Statement ;
    }
}
```

Switch → When we want to use only one case out of multiple cases.

Increment & decrement operator?

→ Increment → To ↑ the value of variable by 1.
→ Decrement → To ↓ — — — — — — —

Learn →

```
int x = 10
int y = x++; // output to be like → x = 11, y = 10
int z = ++x; // x = 11, z = 11.
```

Loops - Statements which allow execution of a code repeatedly.

Types of Loops →

While → Until give cond. is satisfied.

Syntax →
```
while (cond.) {
    statements;
}
```

For →
```
for (initialization; cond.; updation) {
    statements.
}
```

do while →
```
do {
    statements
}
while (<>=)
```

→ Jumping statements →

(1) Break
(2) Continue.
(3) return.

Type casting (changing data type and not value)

```
float b = 87.44
cout << (float) a/34
cout << (int) b;
```

→ Result → 1.6, 87.44

@ What is Array?

→ Stores multiple value in single variable with cont. memory location.

```
data-type arr-var [size];
```

Types → • 1D Array → int arr[size] = {1, 2, 3, 4, 5}
• 2D Array → int arr2d[2][3] = { { 1, 2, 3 }, { 4, 5, 6 } }
2 rows, 3 values

@ What is String? (Used to print a word, eg string name = "Rohan")

→ String is 1-D array of characters terminated by null character.

Syntax - datatypes str-var [size];
char str[6];
char str {"pulkit"};

p	u	l	k	i	t	\0
0	1	2	3	4	5	6 → null

String pre defined function :-

- Str.len() → 6 (length)
- Str.cpy() → (copy to another string) (char name)
- str.rev() → tiklup (reverses the word characters)
- str.cat() → Joining 2 strings
- Str.substr() → To print from here to here

→ To use these add header file # include <string.h>

→ Declaring strings :-

→ char hi[6] = {'H', 'R', 'L'};
cout << hi;

and

String hi = "Hello";
cout hi;

strlen →

```
int main () {
    char str[] = "Pulkit";
    cout << str << endl;
    int r = strlen(str);
    cout << r;
}
```

Result = Pulkit
6

strrev →

```
strrev (str);
cout << str
```

→ Result → tikluP

str cat →

```
~~strcat~~ char str[] = "Pulkit";
char str2[] = "Gupta";

strcat(str, str2);
cout << str;
}
```

Result → Pulkit Gupta

strcpy →

```
char str[] = "Pulkit";
char str2[20];

strcpy(str2, str);
cout << str2;
}
```

Result → Pulkit

substr →

```
string name = "Harry"
cout << name
cout << name.substr(0, 2) → Ha
cout << name.substr(2, 3) → rry
```

(1, 1) → a

(1, 2901) → arry

↳ any large

(1,2) → 1 se start krke huge, 2 tak.

Q What is Pointer?

→ Pointer is a variable that can be to hold adress of another variable.

Note - • & - Adress of operator
• * - value of address operator (to see value of pointer address)

Syntax → data type * var-name;
int *p, a=10;

⇒ int *p;
int a=10;
p=&a;
cout << &a;
cout << p;
cout << a << endl << *p.
Result → (address), address, 10, 10.

Q Function →

→ Block of code that runs only when it is called. Idea is to perform some commonly performed tasks in program to make a function.

```
void max (int x, int y)
{ if (x>y)
      return x;
  else
      return y;
  }
int main() { int a=10, b=20
             int m = max (a,b);
             cout << m.
             return 0. }
```

DATE: / /

@ What is Storage Class?

→ Storage Class defines the scope and lifetime of variable and function.

Types →
(1) Auto
(2) Static
(3) Register
(4) extern (Global)

Storage Class	Memory	default value	Scope	Life time
Auto	RAM	Garbage	within block	Till block active
Static	RAM	0	within	till term. of Progr.
extern	RAM	0	anywhere	"
Register	Register	Garbage	within	Till block is active

Code →

```
int a; // global
int main() {
    int b; // auto(local)
    static int c; // static
    register int d; // register

    cout << a;
    cout << b;
    cout << c;
    cout << d;
    return 0;
}
```

Result ⇒ 0 0 0 1

@ What is Structure?

→ Structure is a user-defined datatype which are used to store dis-similar types of value.

Note :- Size of structure equals to sum of all structure member size

Syntax → struct structure name
{
// members
};

@ Union →

Note :- Size of union is equal to it's biggest no.

Syntax :- Union union-name
{
// members
};

—

Code → using name space std;
struct str
{
int marks;
float avg;
double salary;
};

DATE: / /

```
union ~~std~~ stu 2
{
    int mark
    float avg ;
    double salary;
} ;
int main()
{
    ~~struct~~ stu s ;
    union ~~std~~ stu2 s2 ;
    cout <<"Structure size" ~~<<~~
        << size of (s) ;

    cout << size of (s2) ;
    }
```

Result = 16 → For stu s. (struct wala
(now → 4 bytes + 4 bytes + 8 bytes
8 → For stu s2 (~~max~~ union wali)
Now → Max double ka 8 hai.

★ → Earlier, there was less memory so union was used, now for more memory, struct is used.

★ → Union takes more time to produce output because elements with lower bytes have to use of max bytes only. But in struct, each has it's own bytes.

Structure eg 1.

```
# including <iostream>
using namespace std;
struct stu {
    int roll
    char name [20];
    float marks;
    };

int main() {
    struct ~~stu~~ stu s;
    cout << " roll no."
    cin >> s. roll
    cout << " student name"
    cin >> s. name.
    cout << " marks";
    cin >> s. marks;

    cout << s. roll << s. name << s. marks;
    return 0.
    }
```

Result 1 eg 1 10

Rukkh

400

Union eg →
With same code as structure on left page,
struct ~~wit~~ ki jagah union mein →
Input → 16 Enter 400,
Result → 169082409 400

So for perfect result, put cout << s.size just after cin.

Q What is enum datatype?
→ Enumeration is a user defined ~~ka~~ name that consist of integral constant.

Syntax →
Use it if we have collection like month, week & month
enum enum_name { value1, value2, --- value n

Accessing enums →

```
int main() {
        enum gender {
                male, female, others
                };
        cout << male  → // Result => 0
        cout << female → // Result => 1
        // other technique to cout →
            gender str = others;
            cout << str;
        }
```

Q What is Macro?

→ Macro is a piece of code in a program which is given some name. Whenever the name is used, it is replaced by contents of the macro.

Defined by # define

Syntax →

define macro_name content

Types →

- Object like macro → Variable, object, value
- function like macro → ( ) (in)

Code →

Obj. macro :-

```
# include <iostream>
# define num 10
using namespace std;
int main()
{
    int i;
    for (int i=1, i<=10; i++)
    {
        cout << num * i << endl;
    }
}
```

Q. Function macro →

Eg- # define max(a, b) ((a > b)?, a : b) [Jyada bada banaya?]

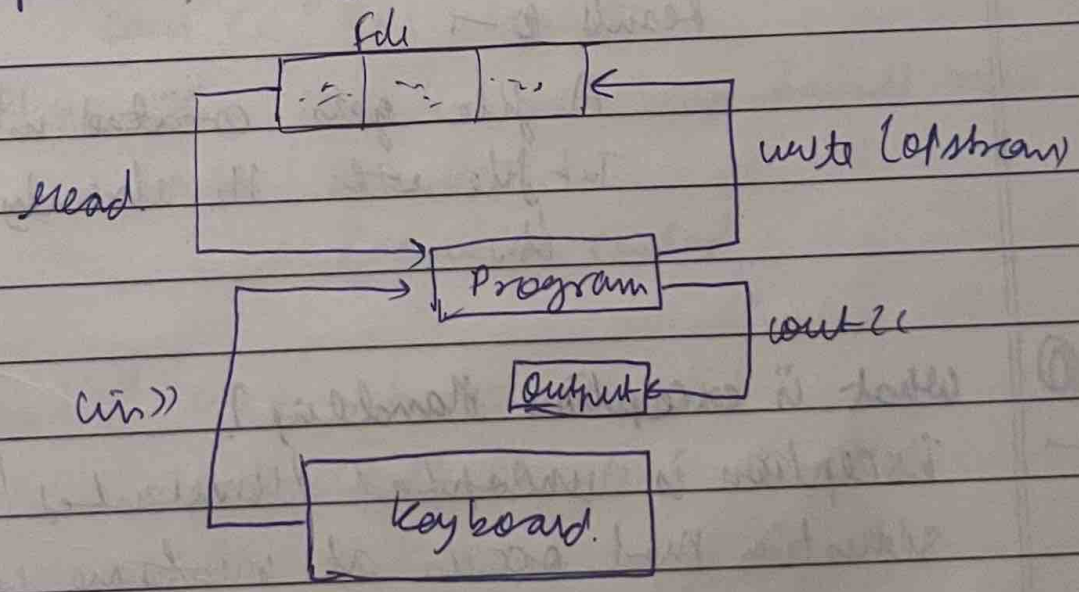
min -

```
int main() {
        cout << max (,  )
          - -  max (,  )
                  |
                many as we need

    }
```

Q. What is file Handling?

→ File Handling is mech so that we can store the output of program in no file and we can perform many operations on the data present in a file.



File operations →

Create → ofstream
write
read → ifstream

Create a file :-

```
# incl. <iostream>
# include <fstream>
using namespace std
int main ()
{
    ofstream newFile("C:\Users\Win10\Desktop\
                                    cpp. txt");
    cout << " file Created..!"
newfile << "Hi"; newFile.close(); ~~newFile.close();~~
}
```

Result :-

A file gets created in Desktop of Txt file with Hi already printed on it.

Q What is exception handling?

→ Exception is unexpected / unwanted / abnormal situation that occur at runtime called exception

→ Agar a/b ka code bnaya rahe hai, hoh wanna 10/0 kiya toh error aaye Jo bhi code ke after hoga wo execute nahi hoga toh tabhi aaye ka code execute hoye isliye, try catch use hoyega :-

```
int main()
{
  cout << " Execution starting "
  int a, b, c;
  cout << " Enter 2 numbers "
    cin >> a >> b;

  try
  {   if(b==0) throw b;
      c = a/b
      cout << " Result : " << c;
  }
  catch (int x)
  {
    cout << " can't divide "
  }
  cout << " Exc. ended "
```

Result → If a = 10, b = 5, it will run and print Result : 2
Execution ended

But if a = 10, b = 0, then,
can't divide by 0.
Exc. ended...

→ Std - Bucket of features
→ Swap - int a=1, int b=2. int c=a, a=b, b=c, cout<<a
→ Snake case - file_cpp | File Size - Pascal Case, | rank[1] → Dimension of array
→ ~~file_size - Camel~~ | ~~iFileSize → Hungarian Notation~~ | Bool Hi = true → 1

Definations

Basics List

- **Program** - Set of statements in a lang. for comp. to execute
- C++ → Semi object oriented, Bjarne, 1979, Bell lab.
  → Case sensitive → Co rectly Classes → Portable fun
- ~~Main - Entry point~~
- Starting entry is main(). (Power button of program)
- First.cpp → Preprocessor → Compiler → First.txt → Linker → First.exe → Loader → First.cpp and Runs.

Data type - Defines the type of value a variable will store.

Primitive - int (4), char (1), float (4), double (8), bool (1), void
Derived - Array, String, Pointer, Reference
User defined - Structure, Union, Class, Enumeration

Variable - Is the name of memory location where data is stored.
- Case sensitive, • Start with (A-Z) (a-z), (_) underscore
- No spaces allowed in between.

Keyword - Reserved word whose meaning is already defined in the compiler.
~ We can't use ~~key~~ Constant, variable as keyword
~ Must be in lower case. 48 + keyword.

Const - Fixed value which can't be changed in run time

Identifier - Name used to define variable, class, function, etc
eg - int aman = 10

Operator - Symbols used to perform mathematical func in C++.

Assignment (=, +=, -=)
Arithmetic (+, -, /, *)
Ternary (?, :), Bitwise (&, |, ^, ~, >>, <<)

PAGE EDGE
DATE: / /

Loops - Statements which allow execution of code repeatedly
For, while, do while.

Array - ~~Stored~~ Stores multiple values in single variable when cont. memory location [datatype arr-var [size]]

String - 1-D array of characters terminated by null characters. [Used to store word like, string name = "Pulkit"]
→ Syntax → datatype str-var [size] → Eg char hi
→ Predefined () - strlen, strcpy, strrev, strcat. Puneet + Gupta
↳ To add man <string.h> header file

Pointer - Datatype used to store address of another variable. & - address, * value of address operator.
→ Synt - datatype * var-name ; (int *p)

Function - Datatype ~~that~~ is which code runs only when it's called. Idea is to reduce some commonly performed task
Eg - max function

Storage Class - Scope and lifetime of variable & function
Types - Auto, Static, extern, register

Structure - User defined D.T. to store diff similar type of value. Size of structure = Sum of all structure member size

Union - Size = ~~the~~ Highest ~~of~~ among all members in union

Union takes less memory while structure takes high. Structure takes more time to execute.

Enum - Used defined name that ~~consists~~ consists of integral constant
→ Enum . enum-name [value], --- value N]