CAP444
OBJECT ORIENTED PROGRAMMING USING C++

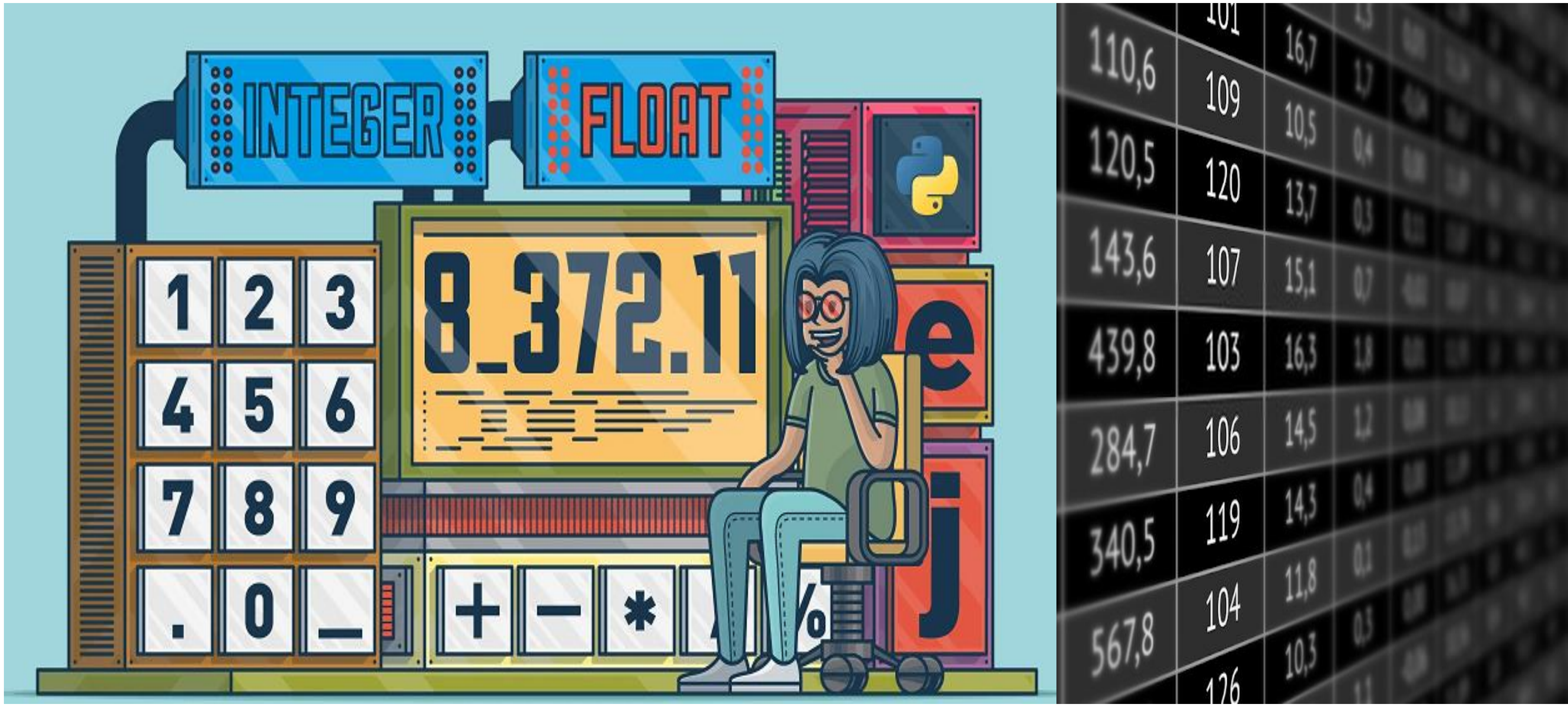
Created By:
Kumar Vishal
(SCA), LPU

Topics Covered

Principles of OOP :

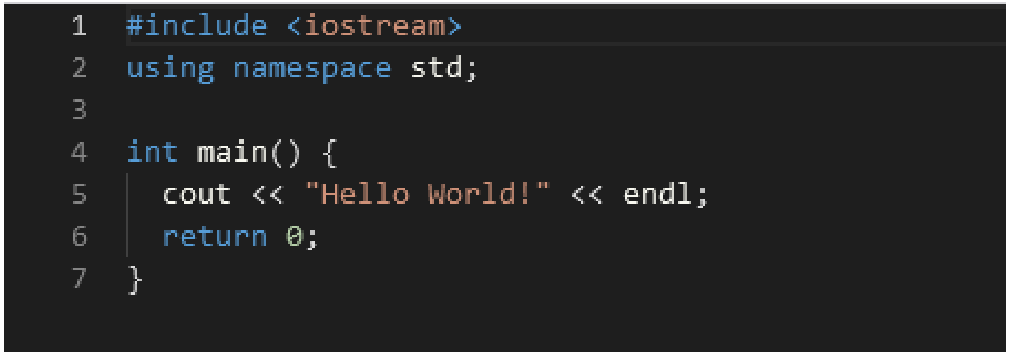
- basic concepts of object oriented programming

Basic concepts of object oriented programming



Programming Structure of C++

- Document Section
- Preprocessor Statement with namespace
- Global Declaration
- Class Definition
- Main Function



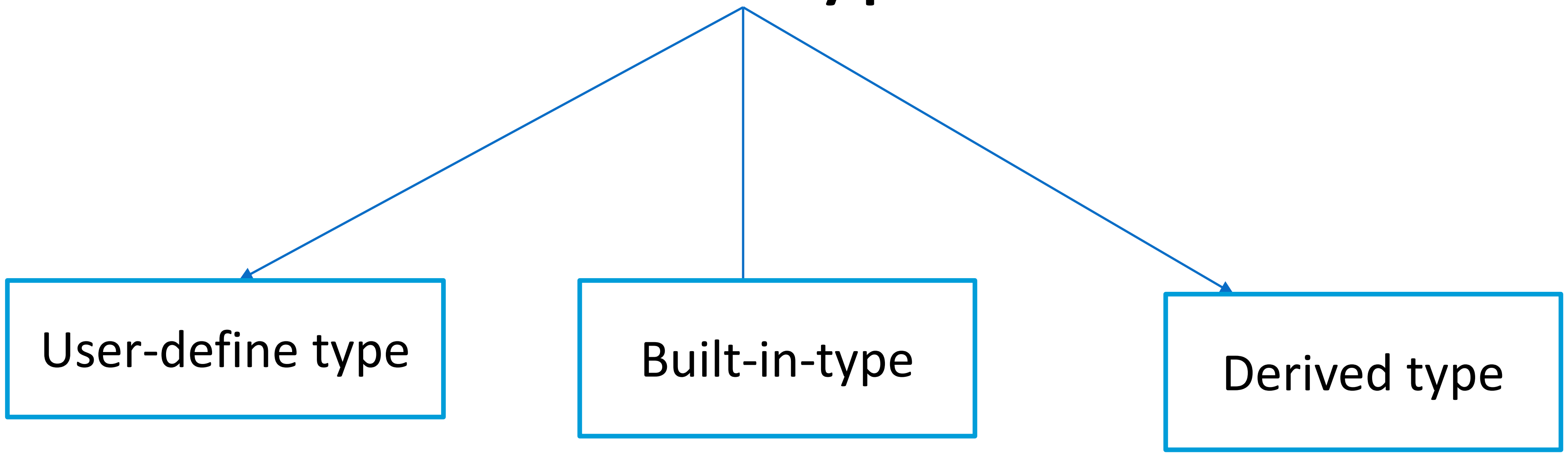
```
#include <iostream>
using namespace std;

int main() {
  cout << "Hello World!" << endl;
  return 0;
}
```

using namespace std;

- ✓ It is known that “std” (abbreviation for the standard) is a namespace whose members are used in the program.
- ✓ So the members of the “std” namespace are cout, cin, endl, etc.
- ✓ This namespace is present in the iostream.h header file.

Data types



- Class
- Structure
- Union

- Integer
- Character
- Boolean
- Floating Point
- Double

- Function
- Array
- Pointer

Primitive Data Types: These data types are built-in or predefined data types and used to declare variables.

Primitive data types available in C++ are:

- Integer(int)
- Character(char)
- Boolean(bool)
- Floating Point(float)
- Double Floating Point(double)

Derived Data Types: The data-types that are derived from the primitive or built-in datatypes are referred to as Derived Data Types.

These are:

- Function
- Array
- Pointer

Abstract or User-Defined Data Types: These data types are defined by user itself.

- Class
- Structure
- Union
- Enumeration or Enum

Data type	Size(in byte)	Range
char	1 =8 bits (2^8)	-128 to 127 or 0 to 255
unsigned char	1	0 to 255
signed char	1	-128 to 127
int	4=32 bits (2^{32})	-2,147,483,648 to 2,147,483,647
short int	2	-32,768 to 32,767
unsigned short int	2	0 to 65,535
unsigned int	4	0 to 4,294,967,295
float	4	
double	8	
long double	12	

We can display the size of all the data types by using the sizeof() operator

Memory representation

1 Byte= 8 Bits

128	64	32	16	8	4	2	1
0	1	0	0	0	0	0	1

A(65)=01000001

char ch=65;
Or
char ch='A'

Char is occupying 1 Byte memory

How to find out range?

For Signed data types:

1.) calculate total number of bits

2.) Calculate -2^(n-1) for minimum range

3.) Calculate (2^(n-1))-1 for maximum range

Unsigned Data Types:

1.)Find number of bits

2.)minimum range is always zero for unsigned data type

3.)for maximum range calculate 2^n-1

Example:

Char : 1 byte: 8 bits=n

Signed: $-2^{(8-1)}$ to $(2^{(8-1)})-1$

=-128 to 127

Unsigned:

0 to $2^{(8)}-1$

=0 to 255

What will be output?

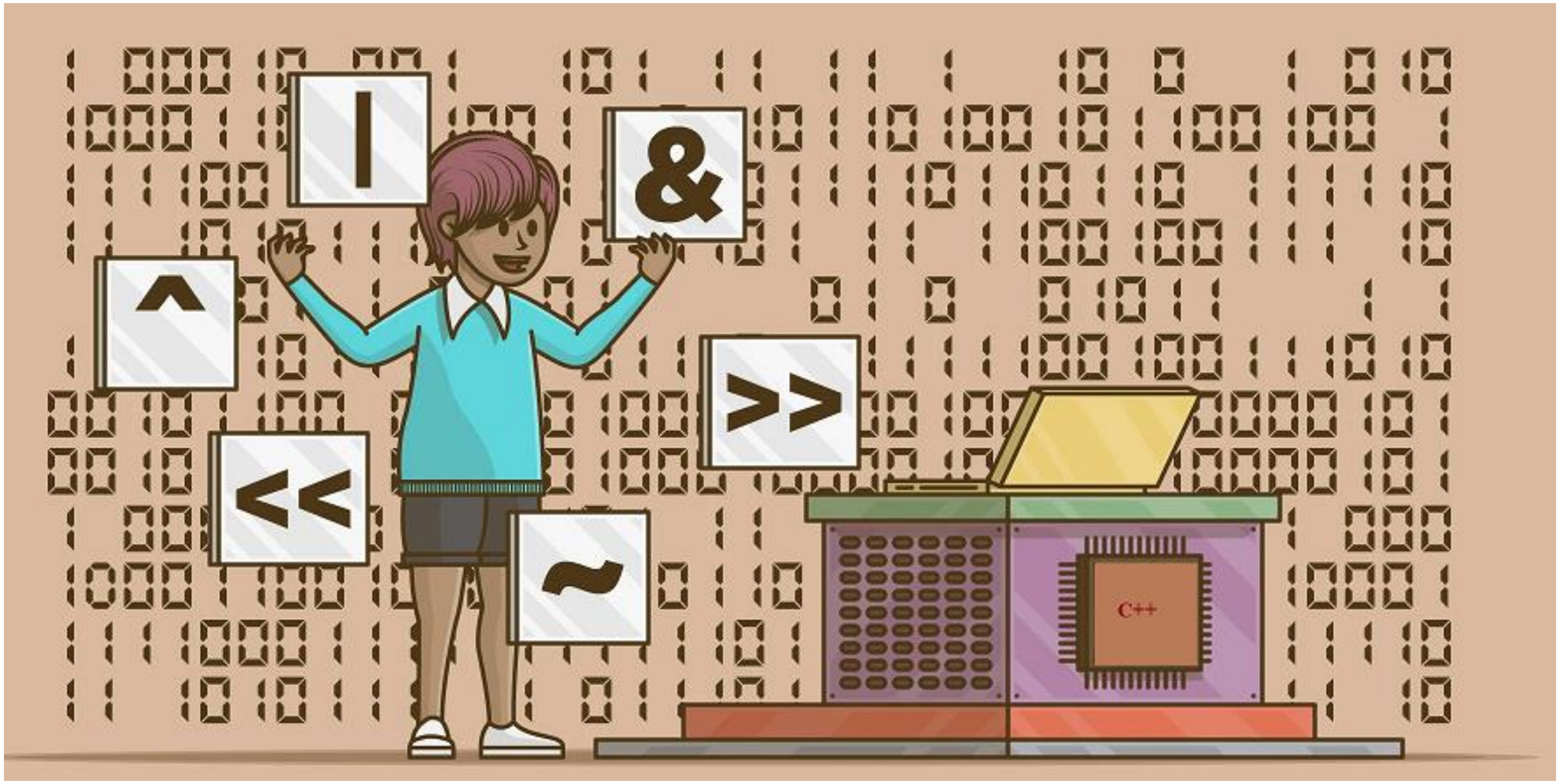
```
#include <iostream>
using namespace std;
int main()
{
   int num=2147483648;
   cout <<num<< endl;
   return 0;
}
```

A. 2147483648
B. - 2147483648
C. Error
D. None

Data type modifiers are:

- Signed
- Unsigned
- Short
- Long

Today we are going to learn about……?



Operators

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Bitwise operators
- Increment /decrement operators
- insertion operator/ extraction operator

Arithmetic operators

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y

Assignment Operators

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3
>>=	x >>= 3	x = x >> 3
<<=	x <<= 3	x = x << 3

Comparison operators

Operator	Name	Example
==	Equal to	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical operators

Operator	Name	Description	Example
&&	Logical and	Returns true if both statements are true	x < 5 && x < 10
||	Logical or	Returns true if one of the statements is true	x < 5 || x < 4
!	Logical not	Reverse the result, returns false if the result is true	!(x < 5 && x < 10)

Bitwise operators

Operator	Description
&	AND Operator
|	OR Operator
^	XOR Operator
~	Ones Complement Operator
<<	Left Shift Operator
>>	Right Shift Operator

AND Operator (&)

If both side bit is on result will be On

a	b	a & b
0	0	0
0	1	0
1	0	0
1	1	1

Steps to solve:-

- **a = 12 (find binary form:1100)**
- **b = 25 (find binary form:11001)**

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
			1	0	0	0	8

a & b=

01100 (12)

11001   (25)

01000   (8) Ans.

What will be output?

```
#include <iostream>

using namespace std;

int main()
{
   int a=20;
   int b=25;
   cout<<(a&b);
return 0;
}
```

A. 15
B. 16
C. 20

OR Operator (|)

If any side bit is on result will be On

a	b	a | b
0	0	0
0	1	1
1	0	1
1	1	1

Steps to solve:-

- **a = 12 (find binary form:1100)**
- **b = 25 (find binary form:11001)**

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
		1	1	1	0	1	29

a | b=

01100 (12)

11001 (25)

11101 (29) Ans.

What will be output?

```
#include <iostream>

using namespace std;

int main()
{
    int a=20;
    int b=15;
    cout<<(a|b);
return 0;
}
```

A. 31
B. 32
C. 22
D. 32

XOR Operator (^)

If both side bit is opposite result will be On

a	b	a ^ b
0	0	0
0	1	1
1	0	1
1	1	0

Steps to solve:-

- **a = 12 (find binary form:1100)**
- **b = 25 (find binary form:11001)**

How to find Binary:

64	32	16	8	4	2	1	
		0	1	1	0	0	12
		1	1	0	0	1	25
		1	0	1	0	1	21

a ^ b=

01100 (12)

11001   (25)

10101   (21) Ans.

Left Shift Operator(<<)

a=10 (1010)

a<<1

1010.0

10100(20) Ans.

a<<2

1010.00

101000(40) Ans.

Right Shift Operator(>>)

a=10 (1010)

a>>1

1010.

101(5) Ans.

a>>2

1010.

10(2) Ans.

What will be output?

```
#include <iostream>
using namespace std;
int main()
{
  int a=15;
  cout<<(a>>1);
return 0;
}
```

Options:

A. 5
B. 6
C. 7
D. 8

Increment/Decrement Operator

++: Increment

++x

--: Decrement

--x

```
int main()
{
    int a=10;
    a++;
    cout<<a;
    return 0;
}
```

What will be output?

```
#include <iostream>

using namespace std;

int main()
{
   int a=10;
   int c=a++;
   cout<<c;

   return 0;
}
```

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

int main()
{
   int a=10;
   int c=++a;
   cout<<c;

   return 0;
}
```

What will be output?

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

 int main()
 {
    int x = 5, y = 5, z;
    x = ++x; y = --y;
    z = x++ + y--;
    cout << z;
    return 0;

 }
```

insertion operator(<<):

The cout is used in conjunction with stream insertion operator (<<) to display the output on a console

extraction operator (>>):

The cin is used in conjunction with stream extraction operator (>>) to read the input from a console.

Control structure

- Conditional structure: if and else
- Selective structure: switch case
- Iteration structures (loops): while, do while, for
- Jump statements: break, continue, goto

if and else

```
if(condition)
{
//Statements(execute when condition true)
}
else
{
//Statements(execute when condition false)
}
```

Switch …case

For menu options:

```
switch(choice)
{
case 1:
break;
default:
}
```

What will be output?

```
#include <iostream>
using namespace std;
int main()
{
   int a=10;
  switch(a)
  {
     case 10:
     cout<<“Hi";
     case 11:
     cout<<“Hello";
  }
   return 0;
}
```

A. Hi
B. Hello
C. HiHello
D. None

While loop

The syntax of a while loop in C++ is −

```
while(condition)
 {
   statement(s);
}
```

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

int main ()
 {
     int a = 10;
     while( a < 20 )
     {
     cout<< a << endl;
     a++;
        }
   return 0;
}
```

Do While loop: at least one time will be execute

The syntax of a do while loop in C++ is −

```
do {
   statement(s);
}
while( condition );
```

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

int main ()
 {
      int a = 10;
  do
      {
       cout<< a << endl;
      a++;
     } while( a > 20 );
   return 0;
}
```

For loop:

The syntax of a for loop in C++ is –

```
for ( initialization; condition; increment )
{
   statement(s);
}
```

Jump statements: break, continue, goto

break: It breaks the current flow of the program at the given condition.

continue: It continues the current flow of the program and skips the remaining code at specified condition.

goto: It is used to transfer control to the other part of the program. It unconditionally jumps to the specified label.

What will be output?

```
#include <iostream>
using namespace std;
int main()
{
   for(int i=1;i<=5;i++)
   {
      if(i==3)
         continue;
      cout<<i;
   }
   return 0;
}
```

A. 12345
B. 123
C. 1245
D. None

go to Jumping Statement

```
#include <iostream>
 using namespace std;
 int main()
 {
 ineligible:
       cout<<"You are not eligible to vote!\n";
       cout<<"Enter your age:\n";
       int age;
       cin>>age;
       if (age < 18){
               goto ineligible;
       }
       else
       {
               cout<<"You are eligible to vote!";
       }
 }
```

What will be output?

```
#include <iostream>
    using namespace std;
    void print(int i)
    {
        cout << i;
    }
    void print(double  f)
    {
        cout << f;
    }
    int main()
    {
        print(5);
        print(500.263);
        return 0;
    }
```

A) 5500.263

B) 500.2635

C) 500.263

Any Query?