PROGRAMMING FUNDAMENTALS

Let's explore technology
together to live in the future

Checkout more on
https://github.com/Sy-hash-collab

Sy-hash-collab

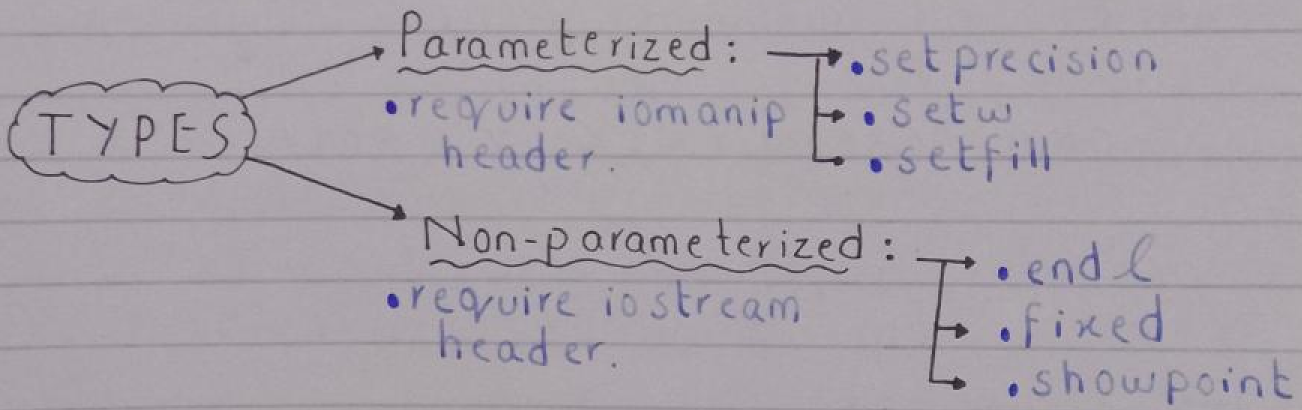
C++ Manipulators

- C++ Manipulators are used to format the output in different styles.
- The manipulators are the most common common way to control output formatting.
- Most of these manipulators are used with 'cout' object.

Syntax:

```
cout << expression        << expression or ... ;
        or manipulator       manipulator
```

- Expression is evaluated
- Value is printed
- Manipulator is used to format output.



TYPES

Parameterized:
- require iomanip header.
- setprecision
- setw
- setfill

Non-parameterized:
- require iostream header.
- endl
- fixed
- showpoint

'setw' Manipulator:

- 'setw' stands for 'set width'
- It is used to display the value of an expression in specified columns.
- The value of expression can be → string. → number.
- Decimal points in floating point values use a space.

All charachters in string uses space.

Value of expression

If value of the expression is less than specified columns the additional coloumns are left blank from left side.

The output automatically uses the required columns if output is larger than the specified columns.

* Must include headerfile iomanip.

The 'setw' manipulator is applied only to the value that is inserted after it. The output is right justified by default.

Syntax:

```
setw (n)
```

The 'n' indicates the number of columns in which the value is to be displayed. It can be an unsigned positive integer constant variable or expression.

```
cout << setw (5) << x << endl;
```

Suppose that the value of a is 33 and the value of b is 7132.

□ represents the space.

Statement | Output

```
cout<<setw(4)<<a<<setw(5)<<b<<setw(4)<<'Hi'.
```

$\underbrace{\square\square 33}_{a}\ \underbrace{\square 7132}_{b}\ \underbrace{\square\square Hi}_{Hi}$

```
Cout<<setw(2)<<a<<setw(4)<<b<<setw(2)<<'Hi'.
```

$\underbrace{33}_{a}\ \underbrace{7132}_{b}\ \underbrace{Hi}_{Hi}$

Note:

If we want to store multiple charach. in a 'char' datatype then we store them as array of charachters.

=> We use 'square bracket []' with variable and then write multiple charachters after assignment operator in " ".

```
char str [] = "OOP using C++";
```

Example:

```
# include <iostream>
# include <iomanip>
using namespace std;
int main ()
{
    int n = 3928,
    double d = 91.5;
    char str [] = "OOP using C++";
    cout << "(" << setw(5) << n << ")" << endl;
    cout << "(" << setw(8) << d << ")" << endl;
    cout << "(" << setw(16) << str << ")" << endl;
}
```

Column:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	3	9	2	8											
				9	1	.	5								
			O	O	P		u	s	i	n	g		C	+	+

Output:

```
(-3928)
(----91.5)
(---OOP-using-C++)
```

'setprecision' manipulator:

- The 'setprecision' manipulator is used to set the number of digits to be displayed after decimal point.
- It is applied to all subsequent floating point numbers written to that output stream.
- Floating point values may be rounded to a number of significant digits.
- By default, the system displays floating point values with six significant digits.

Setprecision → rounds rather than truncates

Setprecision → Trailing zeros are omitted.

21.40 → 21.4

Syntax:

setprecision (n)

=> The 'n' indicates the number of digits displayed after decimal point.

Example:

```
 #include<iostream>
 #include<iomanip>
using namespace std;
int main()
{
 double r, n1=132364 ,n2= 26.91;
  r= n1/n2;
 cout << setpresicion (5) <<r <<endl;
 cout<< setprecision (4) <<r <<endl;
 cout<< setprecision (3) <<r <<endl;
 cout<< setprecision (2)<<r <<endl;
 cout<< setprecision (1) <<r <<endl;
}
```

Output:

```
4.91877
4.9188
4.919
4.92
4.9
```

'setfill' Manipulator:

- The setfill manipulator is used to replace the leading or trailing blanks in the output by specified charachter.
- Requires one parameter to specify the fill charachter. Can a charachter, constant, variable or an integer that represents the fill charachter in ASCII system.

Example 1:

- The manipulators setfill ('*') or setfill (42) will replace the leading or trailing blanks in the output by *.

=> The value 42 is decimal is equivalent to charachter '*' in ASCII.

Example 2:

```
cout <<endl<<setfill ('*')<<"Result
```

will display:

R	E	S	U	L	T

=> The above statement has not specified a field of fixed width to display the string It will appear in field of minimum width.

- It takes exactly six fields to display value.

Example 3:

```
cout <<endl<<setw(15) << setfill ('*') <<"Result";
```

=> The above statement defines field of width 15. It is more than length of value to be displayed

*	*	*	*	*	*	*	R	E	S	U	L	T	S	:

Example:

Output:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    int a = 54786;
   cout << setw(20)<<setfill('*') << a<<endl;
   cout << setw(20)<<setfill('=') << a <<endl;

  }
```

```
***************54786
===============54786.
```

'fixed' Manipulator:

=> The fixed manipulator is used to further control the output of floating-point numbers

- It displays floating point numbers in a fixed decimal format.

Syntax:

```
cout << fixed;
```

Example:

Output:

```
#include <iostream>
#include < iomanip>
using namespace std;
int main()
{
  double a=24.353478 , b=10.353456;
  cout << fixed;
  cout << a << endl;
  cout << b << endl;
```

```
24.353478
10.353456
```

'Showpoint' Manipulator:

=> The showpoint manipulator is used to display the decimal part even if decimal part is zero.

=> Output the numbers with decimal point and trailing zeros.

Syntax:

```
cout << showpoint ;
```

Example:

```
#include <iostream>
#include <iomanip>
using namespace std;
int main ()
{
    float a = 1121.00;
    cout << showpoint ;
    cout << a <<endl;

}
```

Output:

```
1121.00
```

```
int main ()
{
    float a = 11.00;
    cout <<showpoint;
    cout << a <<endl;

}
```

Output:

```
11.0000
```