ICPC Assiut Community

Newcomers Training

DataType and Conditions

ICPC Assiut
community

Training System

- There will be weekly session **Every Saturday**
- There will be a weekly online Practice 3h (**Tue - Wed - Thu**)
- There will be a weekly online contest **(Friday , 7 PM)**
  - **Up Solve** , **Up Solve** , **Up Solve**.
- There will be a weekly sheet.
- After 3 weeks there will be **Filtration**.
- After the end of Training there will be Qualification Contest to join Junior Training.
- Everyone will have **Points** (Attend , Solve problem in sheet , Contest)
- Every session will give **Top 5** in points prize.
- This Training is **Totally Free**.
- Everyone in training will be assigned to Mentor
- Sheet Explain , and join Group in Codeforces.

Points System

- Every Trainee will have A score (Points)
- Every Trainee will gain **20 Points** for every Problem he solve in Practice sheet.
- After every contest the **1st** will take **1000 Points**, the **2nd** will take 90% from **1st**, the **3rd** will take 90% from **2nd**, and so on ...

Your Goals in Training

- Programming Concept (Data Types , Conditions , Loops , Arrays , Functions).
- C++ Language
- How to Search.
- Debug , Test , Fast in Coding .
- Strategy in contest.
- Organize code , Style.
- Learn how to learn
- Build New Network .
- Increase Thinking Skills.
- Building an organized way of thinking in attacking problems.

Rules

- Session **Every week** .
- Last time to attend session after it start within **30 minutes**.
- Should solve **at least 50%** problems weekly sheet.
- **Must** join contest and keep trying to the last minute.
- **Should** attend with your laptop.
- In the end of the training there will be **Certificate** to everyone who solve **at least 80%** of problems.
- Top 10 in Training according to points will take special awards

You should do...

- Register in Codeforces Website.
- Have **CodeBlocks** or any C++ compiler.
- Open the sheet every day and solve.
- Laptop and Internet.

Content

- Intro To Computer science , Programming.
- Importance of Problem Solving and Competitive Programming
- DataTypes and Variables.
- Input / Output.
- Conditions.
- Loops (for, while, do while).
- Arrays 1D and 2D.
- Functions.
- Strings.
- Basic Math.
- Basic Recursion.

Computer

- A **Computer** is a machine or device that performs processes, calculations and operations based on instructions provided by a software or hardware program.

- A **Computer** is a programmable device that can store, retrieve, and process data.

Programming

- **Computer programming** is a way of giving computers instructions about what they should do.

- A **Programming language** is a formal language, which comprises a set of instructions that produce various kinds of output Like C++.

Compiler

- A **Compiler** is a program that **translates** a high level programming language (called source code) into machine language (the target language).

- **Machine language** is a sequence of 0’s and 1’s that the machine (computer) understands and can interpret into instructions.

C++ Compilers

Code::Blocks

Visual Studio®

eclipse

Structure of a Program

```
//  first program in C++

#include <iostream>
int main()
{
  std::cout << "Hello World!";
}
```

Structure of a Program

- **#include <iostream>** (input output stream)

  Known as header **iostream**, that allows to perform standard input and output operations.

- **int main ()**

  A function is a group of code statements which are given a name: in this case, this gives the name "**main**" to the group of code statements that follow.

Structure of a Program

- std::cout << "Hello World!";
- This statement has three parts:

  -First, std::cout, which identifies the standard character output device (usually, this is the computer screen).

  -Second, the insertion operator (<<), which indicates that what follows is inserted into std::cout.

  -Finally, a sentence within quotes ("Hello world!"), is the content inserted into the standard output.
- (;) every statement in c++ end with **semicolon** .

Structure of a Program

Write using **namespace** better

```
#include <iostream>
using namespace std;
int main() {
   cout << "Hello World!";
}
```

Comments

- **Line Comment** : start with **//** and continue until the end of the line.
- **Block Comment** : start with **/*** and end with ***/**.

```
int main ()
{
  cout << "Hello World! "; // prints HelloWorld!
   /*
   Hello world
   c++
   programers
   */
   return 0;
}
```

Data Types

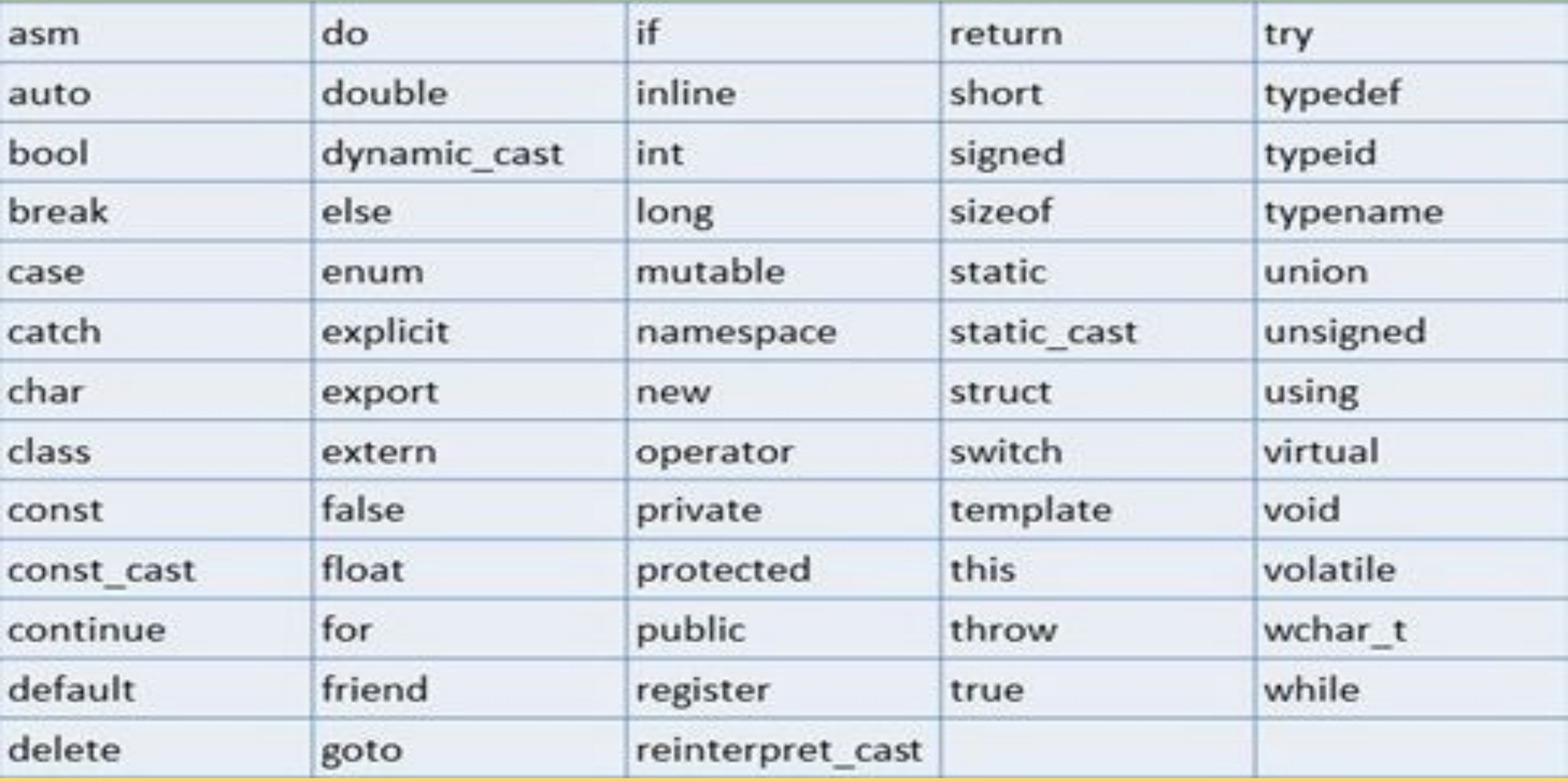
- **int** : Only integers, it`s size : 4 Byte
- **long long** : Only integers, it`s size : 8 Byte
- **float** : Decimals and integers, it`s size : 4 Byte
- **double** : Decimals and integers, it`s size : 8 Byte
- **char** : Symbols, it`s size : 1 Byte
- **bool** : true/false, it`s size : 1 Byte
- **string** : words, it`s size depend on the size of the string

1 Byte = 8 Bits

Declaration Variables

- DataType_Name Varible_Name ;
- Examples :
  - int y;
  - long long z ;
  - char letter ;
  - bool status ;
  - float f1;
  - double salary ;

Reserved Keywords in C++



asm	do	if	return	try
auto	double	inline	short	typedef
bool	dynamic_cast	int	signed	typeid
break	else	long	sizeof	typename
case	enum	mutable	static	union
catch	explicit	namespace	static_cast	unsigned
char	export	new	struct	using
class	extern	operator	switch	virtual
const	false	private	template	void
const_cast	float	protected	this	volatile
continue	for	public	throw	wchar_t
default	friend	register	true	while
delete	goto	reinterpret_cast		

This words can`t use to name a variables or a functions

Initialize Variables

- Datatype_Name Variable_Name = Value ;

  OR
- Datatype_Name Variable_Name ;
- Variable_Name = Value ;

- Examples :
  - int y = 1231; OR int y; y = 1231;
  - long long z = 92233720368547758 ;
  - char letter = 'h' ;
  - bool status = true;
  - float f1 = 3.14;
  - double salary = 15123123123200.64312 ;

Examples

```
#include <iostream>
using namespace std;
int main()
{
    int x;  // Declaration
    int y = 5; // Declaration and Initialization
    float f; // Declaration
    f = 3.14;  // Initialization
    char c = 'h'; // Declaration and Initialization
    bool state = false; // Declaration and Initialization
}
```

String

String Literal: “hello world”

– Ex:

```
string x; // declaration
string z = “hello world“ ; // declaration and definition
```

The size of **z** is : 11 Byte

Simple Program

- Write a program to declare variables :
  val1, ***val2***, ***val3***, ***val4***, ***val5***, ***val6***, ***val7***

- with data types :
  int , ***long long***, ***float***, ***double***, ***char***, ***string***, ***bool***

- After this initialize this variables with values :
  5, ***310000093939***, ***5.34***, ***31.000124***, ***’h’***, ***”ali”***, ***false***

Code

```cpp
#include <iostream>
using namespace std;
int main()
{
    int val1 = 5;
    long long val2 = 310000093939;
    float val3 = 5.34;
    double val4 = 31.000124;
    char val5 = 'h';
    string val6 = "ali";
    bool val7 = false;
}
```

Operator

- **Assignment operator (=)**

  Ex : What is the output of this code ?

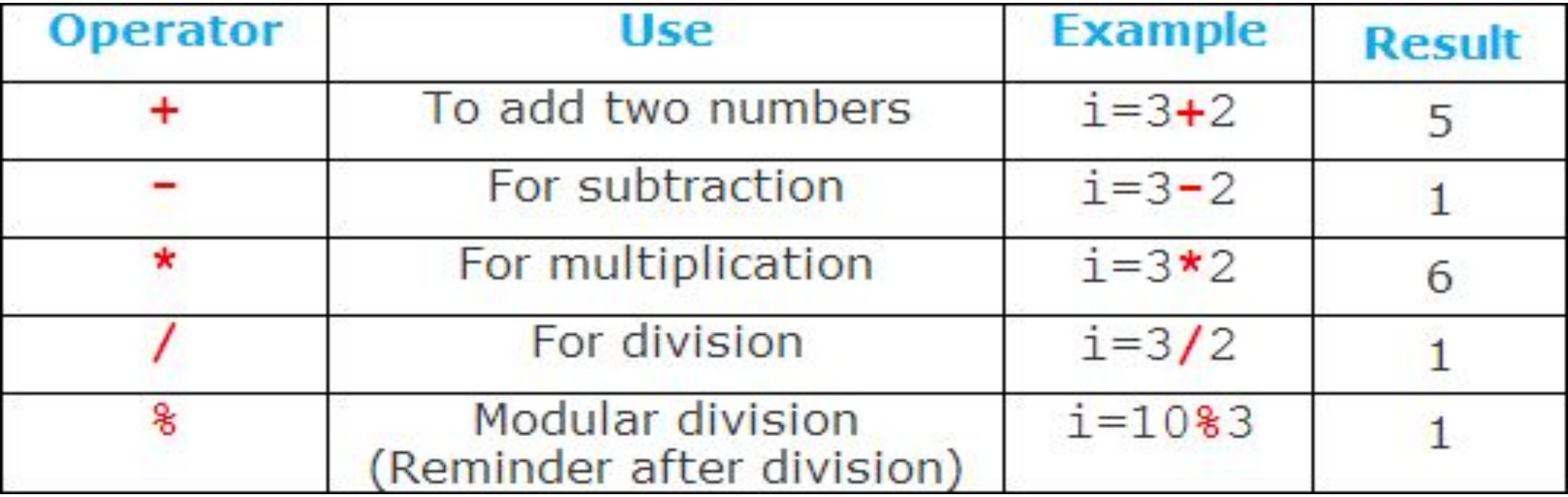
```cpp
#include <iostream>
using namespace std;
int main()
{
    int a, b;
    a = 10;
    b = 4;
    a = b;
    b = 7;
    int x, z;
    x = z = a;
    cout << "a:" << a;
    cout << " b:" << b;
    cout << " x:" << x;
    cout << " z:" << z;
}
```

a = ? , b = ?
a = 10, b = ?
a = 10, b = 4
a = 4, b = 4
a = 4, b = 7
x = ? , z = ?
x = 4, z = 4

so the answer is:

a:4 b:7 x:4 z:4

Operators



Operator	Use	Example	Result
+	To add two numbers	i=3+2	5
-	For subtraction	i=3-2	1
*	For multiplication	i=3*2	6
/	For division	i=3/2	1
%	Modular division (Reminder after division)	i=10%3	1

- Int / int = int
- Int /float = float
- float/ int = float
- int * int = int
- long long * int = long long
- long long * double = double

Modular

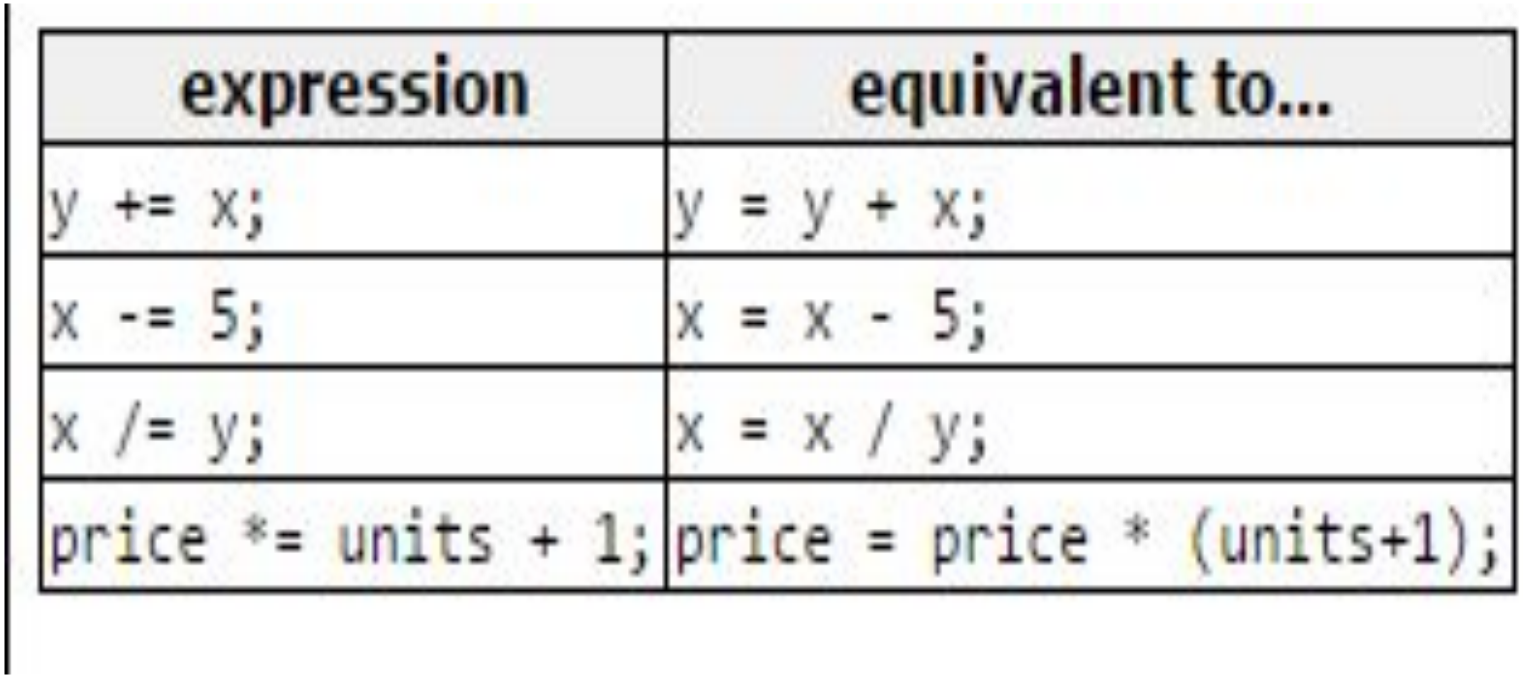
Formula : a % b = a – (a/b) *b ;

EX : int x = 11 % 3 = 11 - (11 / 3) * 3 = 11 - 3 * 3 = 11 - 9 = 2;

Used to :

- Last Digit .
- Multiplication.
- divisibility
- Even Odd.
- Cycle.
- Not Work on doubles.

Compound Assignment



expression	equivalent to...
`y += x;`	`y = y + x;`
`x -= 5;`	`x = x - 5;`
`x /= y;`	`x = x / y;`
`price *= units + 1;`	`price = price * (units+1);`

Important : (**+=**, **-=**, ***=**, **/=**, **%=**)

Example

```
// compound assignment operators
#include <iostream>
using namespace std;
int main()
{
    int sum, sub, x = 1, y = 13;
    sum = x + y;
    sub = x - y;
    int a, b = 3;
    a = b;
    a += 2;         // equivalent to a = a + 2
}
```

Problems

1. Write a program that initialize two variables named **x** and **y** with values **3** ,**5** and print their sum ,and subtract and multiply.

Answer(1)

```
#include <iostream>
using namespace std;
int main()
{
    int x = 3, y = 5, sum, sub, mult;
    sum = x + y;
    sub = x - y;
    mult = x * y;
    cout << "Sum : " << sum << endl;
    cout << "Sub : " << sub << endl;
    cout << "Mult : " << mult << endl;
}
```

Problems

2. Write a program that initialize variable named x with value 123 and print digit (3) and digit (2) and digit (1) (**Hint : use modulo**).

Answer(2)

```
#include <iostream>
using namespace std;
int main()
{
    int x = 123, x1, x2, x3;
    x1 = x % 10;
    x /= 10;
    x2 = x % 10;
    x /= 10;
    x3 = x % 10;
    cout < "First digit : " << x3 << endl;
    cout << "Second digit : " << x2 << endl;
    cout << "Third digit : " << x1 << endl;
    return 0;
}
```

3. Trace this code…

what is the values of **x** and **y** and **z** will be ?

```cpp
#include <iostream>
using namespace std;
int main()
{
    int x = 1, y = 2, z = 3;
    x = y + z;
    y = x * z;
    z = x;
    y = 2;
    x += z;
    z = x % y;
    z /= 13;
    return 0;
}
```

Answer(3)

```
#include <iostream>
using namespace std;
int main()
{
     int x = 1, y = 2, z = 3;
     x = y + z;     x = 5 , y = 2 , z = 3
     y = x * z;     x = 5 , y = 15, z = 3
     z = x;         x = 5 , y = 15, z = 5
     y = 2;         x = 5 , y = 2 , z = 5
     x += z;        x = 10, y = 2 , z = 5
     z = x % y;     x = 10, y = 2 , z = 0
     z /= 13;       x = 10, y = 2 , z = 0
     return 0;
}
```

The Answer is : x = 10 , y = 2 , z = 0

Increment and decrement

- **Increment** : increase value with 1
- **Decrement** : decrease value with 1
- Prefix (++x)
- Postfix (x++)

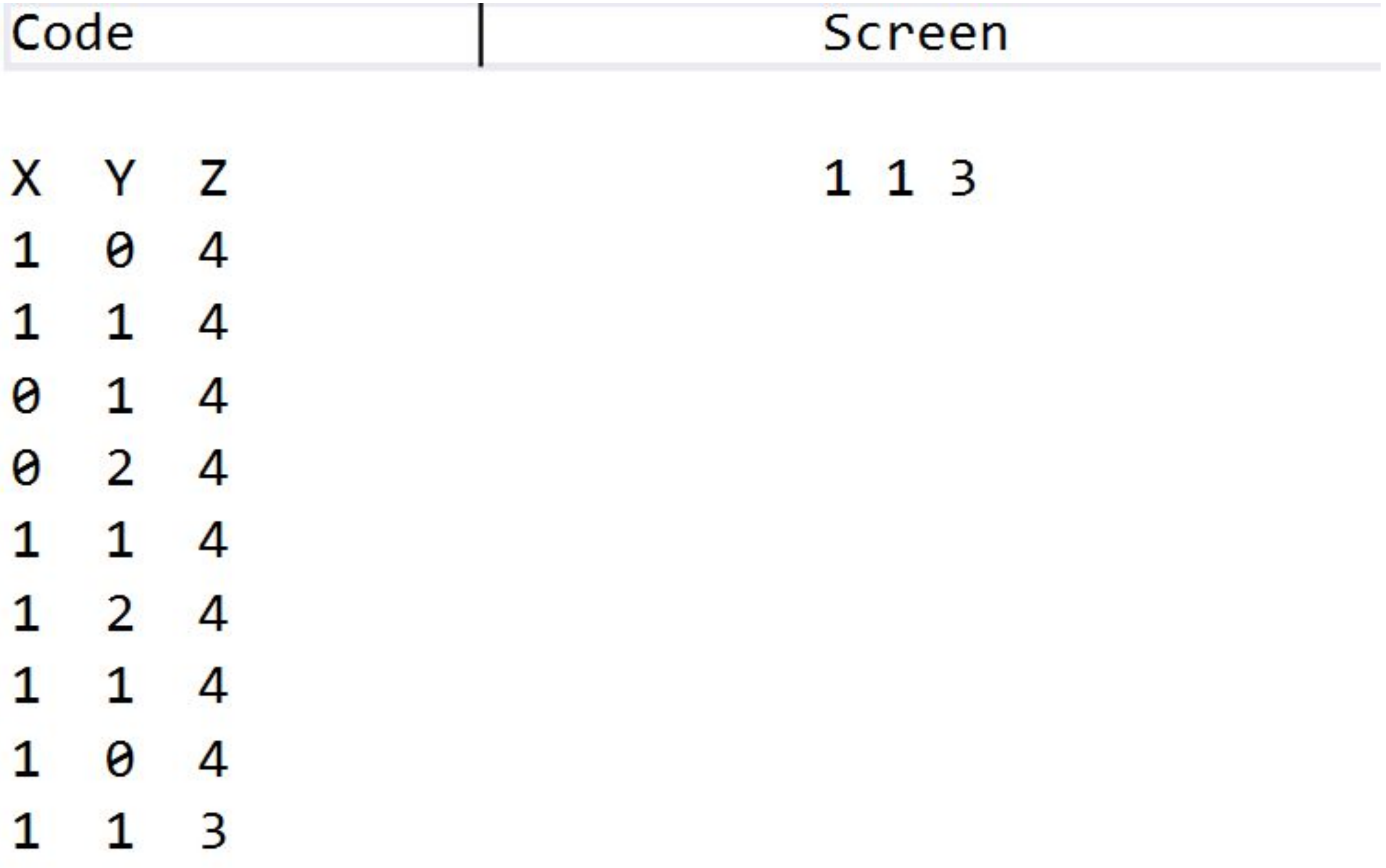
Prefix	**Postfix**

x = 3 ;
y = ++x ;
// x : 4 , y : 4
	x = 3 ;
	y = x++ ;
	// x : 4 , y : 3

Trace this code

```
#include <iostream>
using namespace std;
int main() {
    int x = 1, y = 0, z = 4;
    y++;
    x--;
    y = ++y;
    y = ++x;
    x = y++;
    x = --y;
    x = y--;
    z = x + ++y + (x % 2);
    cout << x << " " << y << " " << z << endl;
    return 0;
}
```

How to trace any code easy



Code			Screen
X	Y	Z	1 1 3
1	0	4	
1	1	4	
0	1	4	
0	2	4	
1	1	4	
1	2	4	
1	1	4	
1	0	4	
1	1	3	

Input/ Output

Cin

For take and input from user

Syntax : cin >> Variable_Name ;

Extraction : (>>)

Ex:

```
int x;
cin >> x;
```

Ex :

```
int a, b;
cin >> a >> b;
```

Same :

```
cin >> a;
cin >> b;
```

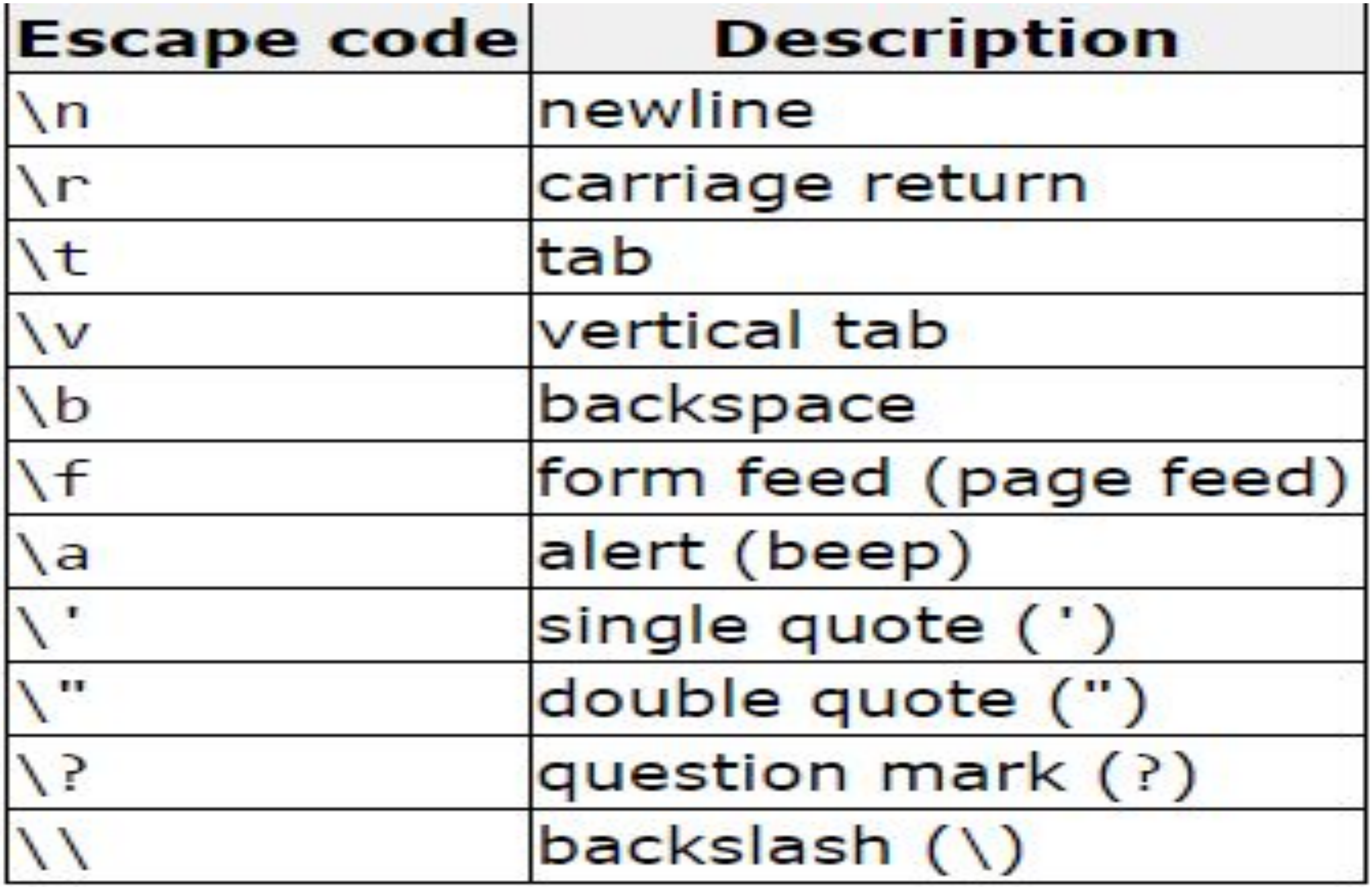
Cout

Insertion : (<<)

Syntax : cout << Variable_Name ;

Ex: cout << x << “ “ << y << endl ;

Another input /output



Escape code	Description
\n	newline
\r	carriage return
\t	tab
\v	vertical tab
\b	backspace
\f	form feed (page feed)
\a	alert (beep)
\'	single quote (')
\"	double quote (")
\?	question mark (?)
\\	backslash (\)

Example

```cpp
#include <iostream>
using namespace std;
int main() {

    cout << "hello world " << '\n';
    cout << "hello world " << '\t';
    cout << "hello world " << '\\';
    cout << "hello world " << '\?';
    cout << "hello world " << endl;
    return 0;
}
```

On screen :

```
hello world
hello world     hello world \hello world ?hello world
```

Problems

1. Write a simple calculator that takes **two numbers** and print its sum, sub and multiply.

2. Write program that take **Name** from user And print Hello and the **Name**.

   Ex :

   input : ahmed

   output : Hello ahmed

3. Write program to calculate this equation : C = x^2 + y * z

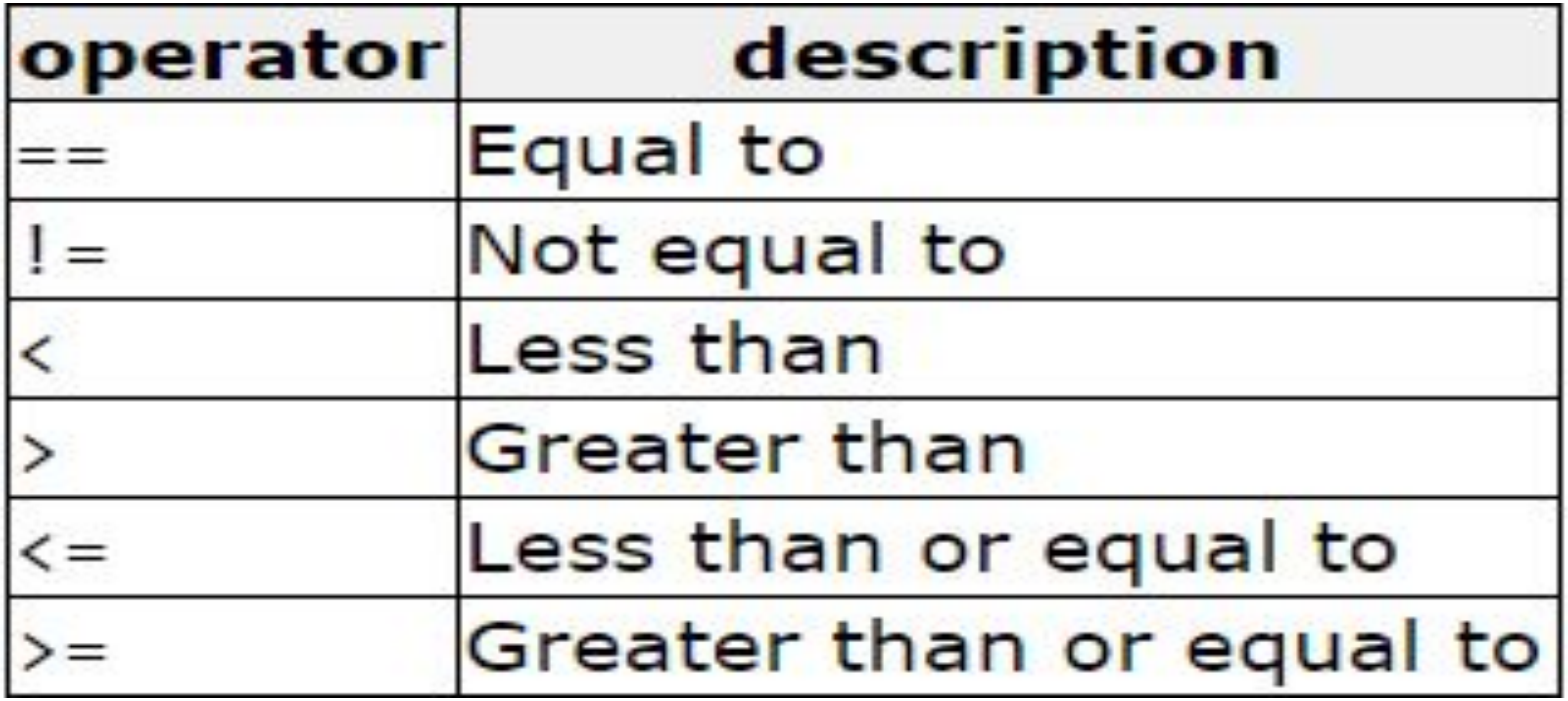
4. Write program that allocate user to enter **two numbers** and swap these numbers and print two numbers after swapping

   Ex : input : 3 6

   output : 6 3

Conditions

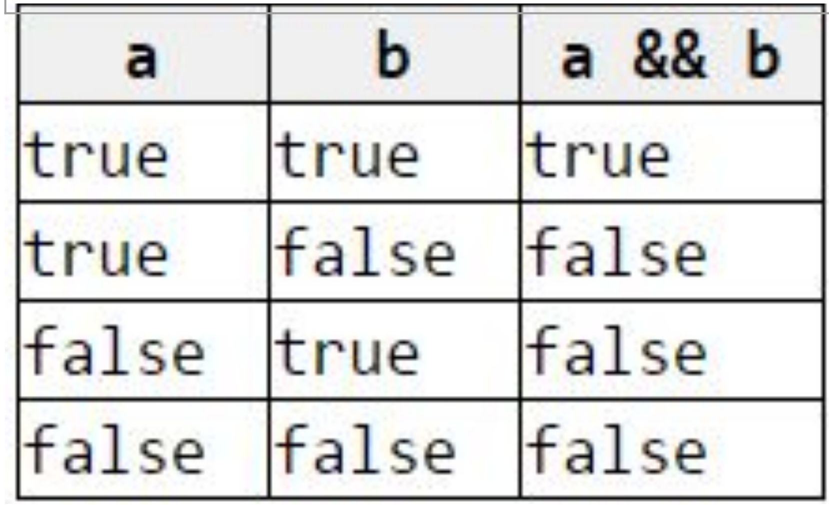
- if (condition) statement



operator	description
==	Equal to
!=	Not equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

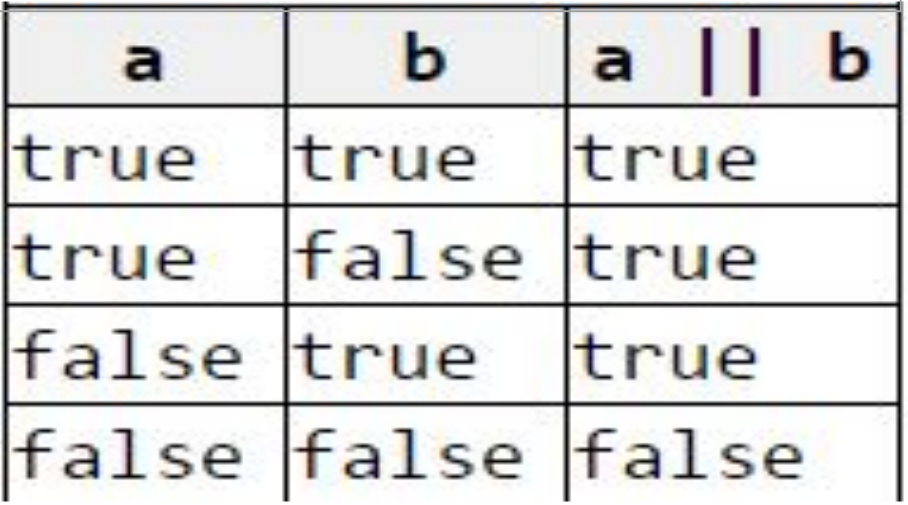
Logical Operators

&& (AND)



a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

|| (OR)



a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

if conditions , nested if

- if (condition)
  ```
  {
         //Statements
  }
  ```

- else if (condition) {
  ```
         //Statements
  }
  ```

- else {
  ```
         //Statements
  }
  ```

Simple code about if , else

```
#include <iostream>
using namespace std;
int main() {
    int number;
    cout << "Enter Number ";
    cin >> number;
    if(number == 100){
        cout << "number is 100" << endl;
    }
    else if(number > 100){
        cout << "number is greater than 100" << endl;
    }
    else {
        cout << "number is less than 100" << endl;
    }
    return 0;
}
```

Your System in Training

- **Study** the topics from videos and tutorials that are in sheet.
    - **When you see any tutorial try their code in your machine to get more understanding.**
- **Solve** sheet's problems and contests.
- **Make** a reference sheet for everything you learn in training.
- **Ask** Mentors or your friend if you don't understand any in the topic.
- You should solve and study **at least 1H every day.**

For more information about **DataTypes** visit this Link

For more information about **If Conditions** visit this Link

Now it's time to practise and solve the problems of Data Types and conditions

DataTypes - Conditions Sheet

Good luck <3