in java we have having one concept named controlled statement

control statements enable control over the program

Basically, there are 3 types of control statements that are

1) Selection
    1) If
    2) If else
    3) If else if
    4) switch
2) Iteration
    1) While
    2) Do while
    3) For
    4) For each
3) Jump statement
    1) Break
    2) Continue
    3) Return

Now required a small code about if else required here

```
int i=10;
if(i<=10)
{
    Syso("Hi");
}
else
{
    Syso("Hello");
}
```

conditional satement inside the parentheses ()

Understanding Separators

Separator	Description
;	Semicolon Terminates statements
,	Comma Separates consecutive identifiers in a variable declaration.
{ }	Braces Define a block of code, for classes, methods and values of arrays
( )	Parentheses Parameters in methods, Precedence in expressions, Control statements
[ ]	Brackets Declare array types, dereference array values
.	Period is used to separate package, sub-packages and classes, variable or method from reference

Seperators:

() ⇨ We can write method parameters & Conditional Statements

[] ⇨ It represents an Array

{} ⇨ These represents a BLOCK of code

; ⇨ It represents END of a statement

, ⇨ It is used for seperation of method parameters & Variables

. ⇨ It represents BELONGS-TO . (It used for calling a method)

Array []

It will collect multiple elements of similar data type.

```
1 package com.pack1 ;
2
3 public class ClassA
4 {
5     void meth1()
6     {
7         int nums[]= {10,20,30,40,50};
8     }
9 }
```

Understanding Java Method

- The only required elements of a method declaration are the method's return type, method name, a pair of parentheses-(), and a body between braces - {}.
- The method declarations have six components, in order:

1. Modifiers :- such as public, private, protected and default.
2. The return type :- the data type of the value returned by the method, or void if the method does not return a value.
3. The method name :- The rules for field names apply to method names as well
4. The parameter list in parenthesis :- a comma is used if you are giving more than one parameter. If there are no parameters, you must use empty parentheses.
5. An exception list :- to be discussed later.
6. The method body, enclosed between braces :- the method's code or logic.

- In general there are two types of methods, User defined and predefined methods.

Returning a value from a method:

Q) When the compiler will be coming back to the calling method?

A) The Compiler will be coming back to the calling method in below mentioned **'3'** scenerios.

⇨ After executing all the statements which were present inside a method

⇨ Whenever the compiler came across ***return*** statement

⇨ If there is an Exception (to be discussed later)

Keypoints:

1) void methods does not need any return statement.

2) Except void methods for any other methods **100%** we need to write a return statement otherwise we will be getting an compile time error

3) return type of a method and returning value of a method should be compatible

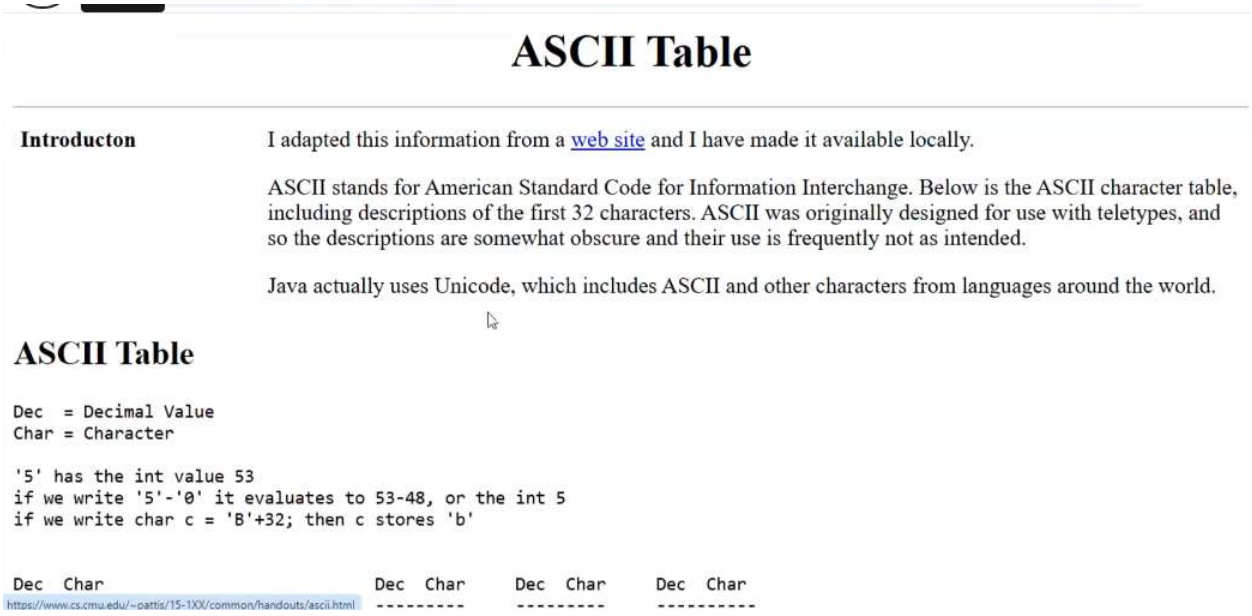
4) return statement **may not** be the last statement inside a method but ***it should be the last executable statement inside a method***

5) Inside a void method we can write a return statement **without returning anything.**

6) For returning a value from a method we are having '4' options, which are listed below

1) void ==> If we dont want to return anything
2) All the '8' Primitive Datatypes [int, byte, short, long, float, double, char & boolean]
3) Classes
4) Interfaces

Whenever the compiler comes across the return statement immediately it will be coming back to the calling method, after the return statement the remaining lines which are present in that method will not be executed by the compiler.



ASCII Table

Introducton I adapted this information from a web site and I have made it available locally.

ASCII stands for American Standard Code for Information Interchange. Below is the ASCII character table, including descriptions of the first 32 characters. ASCII was originally designed for use with teletypes, and so the descriptions are somewhat obscure and their use is frequently not as intended.

Java actually uses Unicode, which includes ASCII and other characters from languages around the world.

ASCII Table

```
Dec  = Decimal Value
Char = Character

'5' has the int value 53
if we write '5'-'0' it evaluates to 53-48, or the int 5
if we write char c = 'B'+32; then c stores 'b'

Dec  Char                           Dec  Char     Dec  Char     Dec  Char
```

https://www.cs.cmu.edu/~pattis/15-1XX/common/handouts/ascii.html

Int and char both are compatible data types

```
'5' has the int value 53
if we write '5'-'0' it evaluates to 53-48, or the int 5
if we write char c = 'B'+32; then c stores 'b'
```

Dec	Char	Dec	Char	Dec	Char	Dec	Char
0	NUL (null)	32	SPACE	64	@	96	`
1	SOH (start of heading)	33	!	65	A	97	a
2	STX (start of text)	34	"	66	B	98	b
3	ETX (end of text)	35	#	67	C	99	c
4	EOT (end of transmission)	36	$	68	D	100	d
5	ENQ (enquiry)	37	%	69	E	101	e
6	ACK (acknowledge)	38	&	70	F	102	f
7	BEL (bell)	39	'	71	G	103	g
8	BS (backspace)	40	(	72	H	104	h
9	TAB (horizontal tab)	41	)	73	I	105	i
10	LF (NL line feed, new line)	42	*	74	J	106	j
11	VT (vertical tab)	43	+	75	K	107	k
12	FF (NP form feed, new page)	44	,	76	L	108	l
13	CR (carriage return)	45	-	77	M	109	m
14	SO (shift out)	46	.	78	N	110	n
15	SI (shift in)	47	/	79	O	111	o
16	DLE (data link escape)	48	0	80	P	112	p
17	DC1 (device control 1)	49	1	81	Q	113	q
18	DC2 (device control 2)	50	2	82	R	114	r

In java we have one concept type casting

Type casting is a process of converting one data type into another data type except Boolean.

Types casting 2 types

1. Implicit type casting
   It means automatically I will be done by the compiler.
2. Explicit type casting
   Programmers(we) need to do that.

Returning a Value from a Method

- A method returns to the code that invoked it when it:
  - Completes all the statements in the method,
  - Reaches a return statement, or
  - throws an exception (covered later),

-Which ever occurs first

Rules:

1. We can declare a method's return type in its method declaration.
2. Inside the body of the method, we should use the 'return' statement to return the value of the return type.
3. Any method declared as 'void' doesn't return any value.
4. void methods don't require any return statement.
5. Any method that is not declared as void must contain a return statement with its corresponding return value.
6. The data type of the return value must match the method's declared retur type.
7. Method return types can be 8 Primitive Datatypes + Void + Class **And** Objects

Note: return statement need not to be last statement in a method, but it must be last statement to execute in a method.

```java
public class ClassA
{
    void meth1()
    {
        System.out.println("meth1() called");
        System.out.println(10);
        System.out.println(20);
        System.out.println(30);
        return;
    }
    int meth2()
    {
        System.out.println("\nmeth2() called");
        return 100;
        //System.out.println("Java is awesome!!"); // C.E because of Unreachable Code
    }
    int meth3(int i)
    {
        System.out.println("\nmeth3() called");
        System.out.println("i value : "+i);
        System.out.println("int & char both are compatable datatypes");
        return 'A';
    }
    String meth4(int i)
    {
        System.out.println("\nmeth4() called");
        if(i<=5)
        {
            System.out.println("If block executed");
            return "Java";
            //System.out.println("hi");// C.E because of Unreachable Code
        }
        else
        {
            System.out.println("else block executed");
            return "Java is awesome";
            //System.out.println("hello");// C.E because of Unreachable Code
        }
    }
    public static void main(String[] args)
    {
        System.out.println("Start");
        ClassA aobj=new ClassA();
        aobj.meth1();
        System.out.println("meth2() is returning : "+aobj.meth2());
        System.out.println("meth3() is returning : "+aobj.meth3('a'));
        System.out.println("meth4() is returning : "+aobj.meth4(1));
        System.out.println("\nEnd");
    }
}
```