- && and || cause short circuit evaluation
- if the first part of p && q is false then q is not evaluated.
- if the first part of p || q is true then q is not evaluated.

```
// example
if (x <= X_LIMIT && y <= Y_LIMIT)
// do something.
```

More flow of control :-

- if-else :

```
if ( boolean-expression)
    statement1;
else
    statement2;
```

- Multiway selection :

```
if (boolean-expression 1)
    statement1;
else if ( boolean-expression2)
    statement2;
else
    statement3;
```

Page No.
Date

While Loops :-

- While loops
  while (boolean-expression)
    statement ; // or statement block
- do-while loop part of language
  do
    statement;
  while (boolean-expression);
- Again, could use a statement block
- break, continue, and labeled breaks
- referred to in the Java tutorial as branching statements

keywords to override normal loop logic

use them judiciously (which means not much)

Attendance question 4 :-

What is output by the following Java code?

```
int x = 3;
double a = x / 2 + 3.5;
System.out.println(a);
```

1) a
2) 5
3) 4.5
4) 4
5) 5.0

Arrays :-

Arrays in Java :-

- Java has built in arrays. native arrays
- arrays hold elements of the same type
  - primitive data types or classes
  - space for array must be dynamically allocated with new operator.

```
Public void arrayExamples ()
{ int [] intList = new int [10];
  for (int i = 0 ; i < intList.length; i++)
  { assert 0 >= i && i < intList.length;
    intList [i] = i * i * i;
  }
  intList [3] = intList [4] * intList [3];
}
```

Array Details :-

- all arrays must be dynamically allocated
- arrays have a public, final field called length
  - built in size field, no separate variable needed.

- don't confuse length (capacity) with elements in use.

- elements start with an index of zero, last index is length - 1

trying to access a non existent element results in an ArrayIndexOutofBounds-Exception (AIOBE)

Array Initialization :-

- Array variables are object variables
- They hold the memory address of an array object.
- The array must be dynamically allocated
- All values in the array are initialized (0, 0.0, char 0, false, or null)
- Arrays may be initialized with an initializer list:

```
int[] intList = {2, 3, 5, 7, 11, 13};
double[] dList = {12.12, 0.12, 45.93};
String[] sList = {"Olivia", kelly",
                  "Isabelle"};
```

Arrays of objects :-

- A native array of objects is actually a native array of object variables.
  - All object variables in Java are pointers!

Array Utilities :-

- In the Arrays class, static methods
- binarySearch, equals, fill, and sort methods for arrays of all primitive types (except boolean) and arrays of objects.
  - overloaded versions of these methods for various data types.
- In the System class there is an arraycopy method to copy elements from a specified part of one array to another.
  - can be used for arrays of primitives or arrays of objects.

Page No.
Date

2D Arrays in Java:-

- Arrays with multiple dimensions may be declared and used

```
int [] [] mat = new [3] [4];
```

- the number of pairs of square brackets indicates the dimension of the array.
- by convention, in a 2D array the first number indicates the row and the second the column.
- Java multiple dimensional arrays are handles differently than in many other programming languages.

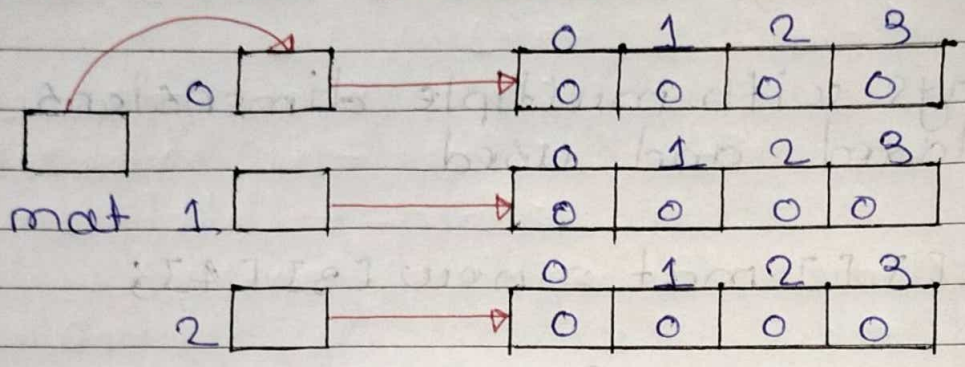
Two Dimensional Arrays :-

row \ column	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
row				

- This is our abstract picture of the 2D array and treating it this way is fine.

```
mat [2] [1] = 12;
```

The Real Picture:-



- mat holds the memory address of an array with 3 elements. Each element holds the memory address of an array of 4 ints.

Arrays of Multiple Dimensions:-

- Because multiple dimensional arrays are treated as arrays of arrays of arrays.... multiple dimensional arrays can be ragged.

- each row does not have to have the same number of columns.

```
int [] [] raggedMat = new int [5] [];
for (int i = 0; i < raggedMat.length; i++)
    raggedMat [i] = new int [i + 1];
```

- each row array has its own length field.

Page No.
Date

Enhanced for Loop :-

- New in Java 5.0
- The for-each loop
- useful short hand for accessing all elements in an array (or other types of structures) if no need to alter values
- alternative for iterating through a set of values
- for (Type loop-variable : set-expression)
  statement
- logic error (not syntax error) if try to modify an element in array via enhanced for loop.

Examples :-

1)
```
Public static int sumListold (int [] list)
{  int total =0;
   for (int i =0; i< list.length; i++)
   { total += list [i];
     System.out.println (list [i]);
   }
   return total;
}
```

2) public static int sumListEnhanced (int [] list)

```
{ int total = 0;
  for (int val : list)
  { total += val;
    System.out.println (val);
  }
  return total;
}
```

Attendance question 5 :-

What is output by the code to the right when method d1 is called?

A. 322 B. 323 C. 363
D. 366 E. 399 ~~F.~~

```
public void d2 (int x) {
    x *= 2;
    System.out.println (x);
}

public void d1 () {
    int x = 3;
    System.out.print (x);
    d2 (x);
    System.out.print (x);
}
```

Ans :-

Page No.
Date

* Attendance question 6 :-

What is output by the code to the right ?

A. output will vary from one run of program to next.

B. 00
8

C. 363

D. 7 then a runtime error

E. No output due to syntax error.

```
int [] list = {5, 1, 7, 3};
System.out.print( list[2]);
System.out.print (list[4]);
```

Ans:→ 7 then a runtime error.

Methods:-

- Methods are analogous to procedures and functions in other languages.
  - local variables, parameters, instance variables
  - must be comfortable with variable scope: where is a variable defined?
- Methods are the means by which objects are manipulated (objects state is changed) much more on this later.
- Method header consists of
  - access modifier (public, package, protected, private)
  - static keyword (optional, class method)
  - return type (void or any data type, primitive or class)
  - method name
  - parameter signature

* Static Methods :-

- The main method is where a stand alone Java program normally begins execution.

- common compile error, trying to call a non static method from a static one

```
public class StaticExample{
  public static void main (String [] args)
{ //starting point of execution
  System.out.println("In main method");
   method1();
   method2(): //compile error;
}
public static void main()
{ System.out.println("method 1"); }
public void method2()
{ System.out.println("method 2"); }
}
```

* Method Overloading and Return:-

- A class may have multiple methods with the same name as long as the parameter signature is unique

- may not overload on return type

- Methods in different classes may have same name and signature
  - this is a type of polymorphism, not method overloading
- If a method has a return value other than void it must have a return statement with a variable or expression of the proper type.

  Multiple return statements allowed, the first one encountered is executed and method ends

* Method Parameters :-

- A method may have any number of parameters
- Each parameter listed separately
- No VAR (pascal), &, or const & (C++)
- Final can be applied, but special meaning
- All parameters are pass by value

* Value Parameters vs. Reference Parameters

- A value parameter makes a copy of the argument it is sent.

  - changes to parameter do not affect the argument

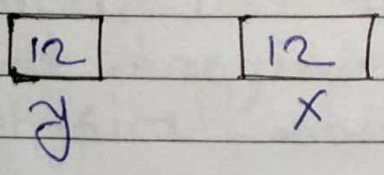
- A reference parameter is just another name for the argument it is sent.

  - changes to the parameter are really changes to the argument and thus are permanent.

* Value vs. Reference :-

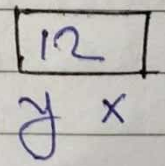
```
// value
void addTo(int x)
{ x += 10; }

void calls()
{ int y = 12;
  addTo(y);
  // y = ?
}
```




```
// c++, reference
void addTo(int& x)
{ x += 10; }

void calls()
{ int y = 12;
  addTo(y);
  // y = ?
}
```



Page No.
Date

* Assertions :-

- Assertions have the form
  assert boolean expression : what to output if assertion is false.

- Example

```
if ( (x<0) || (y<0) )
{ // we know either x or y is <0
  assert x<0 || y<0 : x + " " + y;
  x += y;
}
else
{ // we know both x and y are not less than zero
  assert x>=0 && y>=0 : x + " " + y;
  y += x;
```

- Use assertion liberally in your code
  - part of style guide

Page No.
Date

* Feature: object oriented Programming :-

* object-oriented Programming is a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships.
- Grady Booch.

Terms
- object
- class
- Abstraction
- Encapsulation
- Inheritance

* Object :-

- A thing in a real world that can be either physical or conceptual. An object in object oriented programming can be physical or conceptual.
- conceptual objects are entities that are not tangible in the way real-world physical objects are.

Page No.
Date

- Bulb is a physical object. While college is a conceptual object.
- conceptual objects may not have a real world equivalent. For instance, a stack object in a program.
- A bulb has state and behavior and that is switch on & switch off.

* Attributes and operations :-

- The object's state is determined by the value of its properties or attributes.
- Properties or attributes → member variables and /or data members
- The object's behaviour is determined by the operations that it provides
- operations → member functions or methods

* class :-

- A class is a construct created in object-oriented programming languages that enables creation of objects.

Page No.
Date

- Also sometimes called blueprint or template or prototype from which objects are created

- It defines members (variables and methods)

- A class is an abstraction.

* Abstraction :-

* Abstraction denotes essential characteristics of an object that distinguish it from all other kinds of objects and thus provide crisply defined conceptual boundaries, relative to the perpective of the viewer.
- Grady Booch.

- Abstraction is the process of taking only a set of essential characteristics from something.

- Example
- For a Doctor → you are a patient
  - Name, Age, old medical records
- For a Teacher → you are a student
  - Name, Roll Number / Reg No, Education background.

Page No.
Date

* class and Encapsulation :-

Student	Rectangle	LinkedList
• -roll : long	• -length : int	• -Node
• -name : String	• -width : int	• +addFirst (Node n)
• +display() : void	• +area() : int	• +add (Node n, int pos)
• +read() : bool		• +remove (Node n)

- : private
+ : public.

* Reuse in object oriented Language :-

- Object oriented Languages also implements reuse in the same way that we do in real life.

- Using
  - has-a
  - is-a

- Has-a or composition relationship is implemented by having a class having another class as its member, or rather an object having another object as its member.

  - car has a stereo
  - college has Teachers and students.

- Is-a is implemented through what we call inheritance relationship

* Inheritance :-

* Inheritance defines relationship among classed, wherein one class share structure or behaviour defined in one or more classes.
- Grady Booch

- Defines Is-A relationship between classes
- cat Is-A Animal
- car Is-A Vehicle
- Rose Is-A Flower

* Inheritance Hierarchy :-