**Are functions in java pass by value or pass by reference or both ?**

by default java is pass by value, In java there is no such thing as pass by reference like in C++

In C++ we can do this

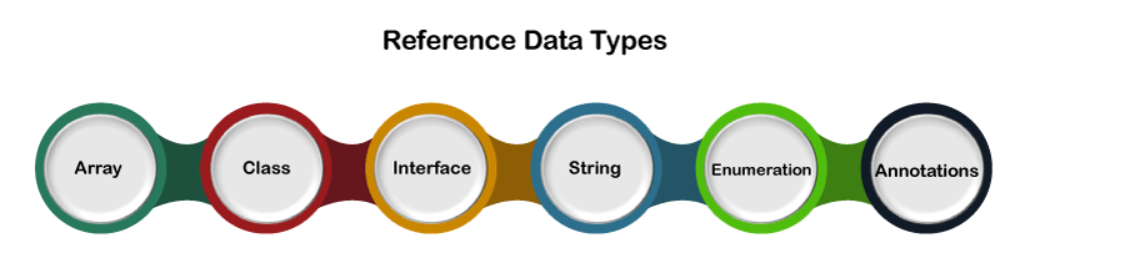


Here even with primitive data types we can pass the address of the variable and we can obtain pass by reference

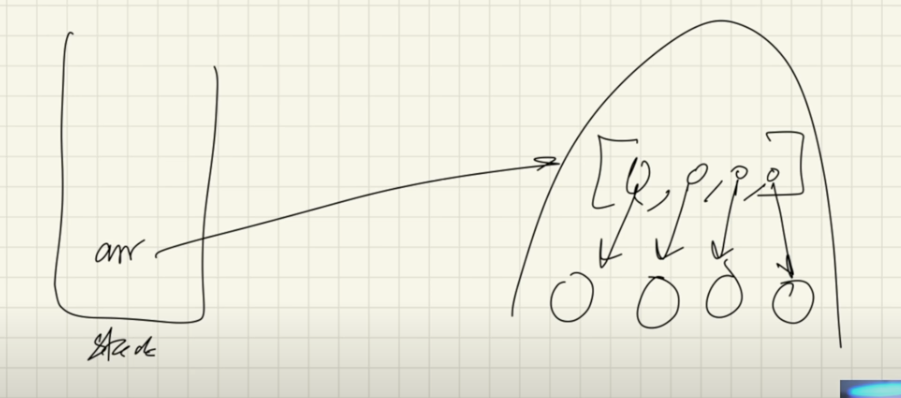
We can not do something like this in Java

**But do not be confused**

In java we have reference variables also, so when we pass those variables through function it may seem like pass by reference but actually that’s because the variable name itself is an reference



**Internally an ArrayList looks like this**



the reference variable of the ArrayList object lives in stack and points to the object in the heap

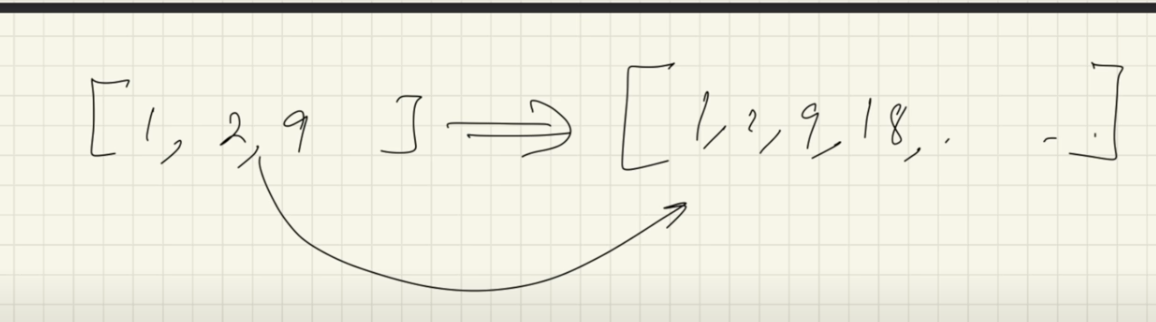
Also in ArrayList primitive data types are not allowed, instead we are only allowed wrapper classes

So each index of the array in the heap points to the object of the wrapper class

**How come we put as many elements in ArrayList object as we want and it is never filled ?**

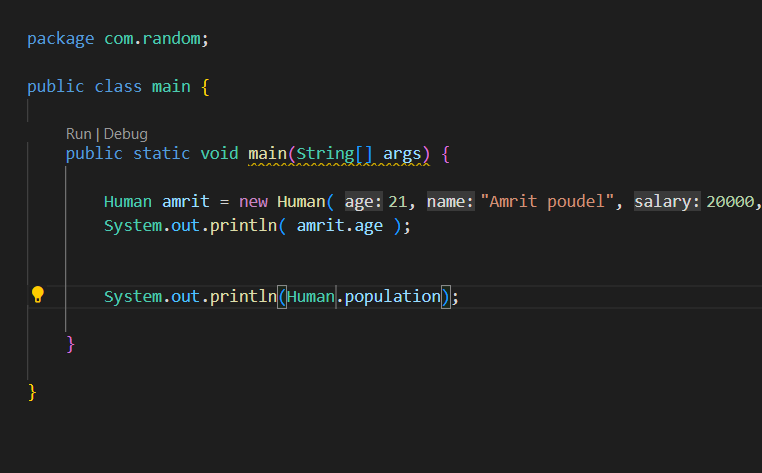
**How is it saying that no size is required to put initially while declaring the ArrayList object like we do for arrays ?**

1. Size is actually fixed internally
2. When the ArrayList fills by some amount, then a new ArrayList of greater size will be created
3. And all the elements from the old ArrayList will be copied in to the new one
4. This all happen internally by Java
5. This whole process has time complexity of O(1)



Inside the main class, we have public static void main ( ) { } method

Why is this method static ?



In Java the program starts to run from the main function first, so if the main function is not static it has to be called via creating object

But how could we create an object if main function( ) itself is the starting point of the program

So it is static, and it can be called via class name directly

What is dynamic method dispatch (DMD) in java?

Why static methods can be overloaded but cannot be overridden?

Why default was introduced in interface?