Arrays

How is it stored in memory?

- One contiguous block in memory
- When array is initialize, the JVM allocate memory for that array (that's why it's size can't be changed)
- Every element occupies the same amout of space in memory (same amout of bytes)

Primitive Type Keyword FacingIssuesO

Type	Size in bytes	Range	Default Value
byte	1 byte	-128 to 127	0
short	2 bytes	-32,768 to 32,767	0
int	4 bytes	-2,147,483,648 to 2,147,483, 647	0
long	8 bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	0
float	4 bytes	approximately ±3.40282347E+38F (6-7 significant decimal digits) Java implements IEEE 754 standard	0.0f
double	8 bytes	approximately ±1.79769313486231570E+308 (15 significant decimal digits)	0.0d
char	2 bytes	0 to 65,536 (unsigned)	'\u0000'
boolean	Not precisely defined*	true or false	false

- When we create array of Objects we store the **reference** to this objects (objects references are always of the same size)
- We cant easily calculate the memory address of and array element based on its index:
  x – array starts memory
  y – size of each element in memory
  I – ith element
  x + i * y
- If we know the index of an element, the time to retrieve the element will be the same, no matter where it is in the array

Retrieve an Element from an Array

1. Multiply the size of the element by its index
2. Get start address of the array
3. Add the start address to the result of the multiplication

No matter how many elements array store it takes only 3 steps to retrieve the desired item

When we know index of element in array, retrieving desired item has constant time complexity, O(1)