Arrays Task

- Write a program using arrays:
  - Create student_names array which holds names of any 5 students, the names will be input by the user
  - Create student_marks array which holds marks of those 5 students, again input by the user
  - Print it in following format
    - Name Marks
    - Ali 50
    - Ahmed 60

Multidimensional Arrays

- Array of Arrays
- Normally we will stick to 2D Array

```
int twoD[][] = new int[4][5];
```

Code Demonstration

```
// Demonstrate a two-dimensional array.
class TwoDArray {
  public static void main(String args[]) {
    int twoD[][]= new int[4][5];
    int i, j, k = 0;

    for(i=0; i<4; i++)
      for(j=0; j<5; j++) {
        twoD[i][j] = k;
        k++;

      }

    for(i=0; i<4; i++) {
      for(j=0; j<5; j++)
        System.out.print(twoD[i][j] + " ");
      System.out.println();
    }
  }
}
```

Allocate second Dimension Manually

```
int twoD[][] = new int[4][];
twoD[0] = new int[5];
twoD[1] = new int[5];
twoD[2] = new int[5];
twoD[3] = new int[5];
```

Alternatives

```
int a1[] = new int[3];
int[] a2 = new int[3];
```

The following declarations are also equivalent:

```
char twod1[][] = new char[3][4];
char[][] twod2 = new char[3][4];
```

This alternative declaration form offers convenience when declaring several arrays at the same time. For example,

```
int[] nums, nums2, nums3; // create three arrays
```

creates three array variables of type **int**. It is the same as writing

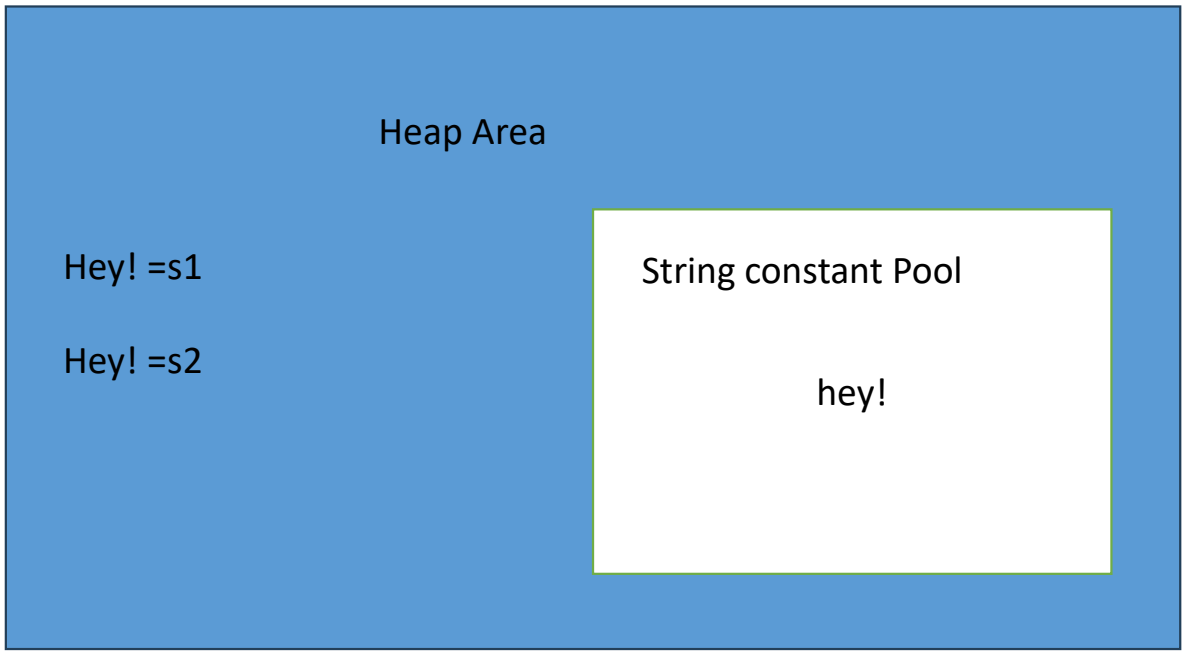
```
int nums[], nums2[], nums3[]; // create three arrays
```

Arrays

- Declaration
- Initialization
- Multi Dimensional Arrays

Strings

- Not a primitive type
- Rather it is an object in java


Heap Area
String constant Pool
hey!
Hey! =s1
Hey! =s2

Copying Arrays

- Copying One Array to Other
- = operator
- Loop to copy

Type conversion

- You assign a value of one data type to another:
  - Two types might not be compatible or might be
- If Data types are compatible:
  - Java will perform the conversion automatically known as Automatic Type Conversion
- If not then they need to be cast or converted explicitly.
  - For example, assigning an int value to a long variable.

Datatype	Bits Acquired In Memory
boolean	1
byte	8 (1 byte)
char	16 (2 bytes)
short	16(2 bytes)
int	32 (4 bytes)
long	64 (8 bytes)
float	32 (4 bytes)
double	64 (8 bytes)

Widening or Automatic Type Conversion

- Automatically done by Java
- When:
  - Two Data Types are compatible
    - Like numeric types
    - Numeric to boolen or char is incompatible
  - Assign the value of smaller dtype to bigger dtype

Byte –> Short –> Int –> Long – > Float –> Double

Widening or Automatic Conversion

```java
// Main class
class GFG {

    // Main driver method
    public static void main(String[] args)
    {
        int i = 100;

        // Automatic type conversion
        // Integer to long type
        long l = i;

        // Automatic type conversion
        // long to float type
        float f = l;

        // Print and display commands
        System.out.println("Int value " + i);
        System.out.println("Long value " + l);
        System.out.println("Float value " + f);
    }
}
```

Narrowing or Explicit conversion

- Larger data type to Smaller Data type:
  - Useful for incompatible types

Error (int 4 bytes, char 2 bytes)

```java
// Java program to illustrate Incompatible data Type
// for Explicit Type Conversion

// Main class
public class GFG {

    // Main driver method
    public static void main(String[] argv)
    {

        // Declaring character variable
        char ch = 'c';
        // Declaringinteger variable
        int num = 88;
        // Trying to insert integer to character
        ch = num;
    }
}
```

```java
// Main class
public class GFG {

    // Main driver method
    public static void main(String[] args)
    {

        // Double datatype
        double d = 100.04;

        // Explicit type casting by forcefully getting
        // data from long datatype to integer type
        long l = (long)d;

        // Explicit type casting
        int i = (int)l;
```