THE UNIVERSITY OF SYDNEY

INFO1113 Week 2 Tutorial

Loops and arrays

Loops and Arrays

As with any programming language there is some form of iteration. Within the `java` language we have access to `for` and `while` keywords which allow you to create loops.

`while` loop

A while loop requires a condition to be true for it to commence and continue iterating. The while loop structure follows `while(`*condition*`)`, after this component is defined, the following code block will continue to be executed.

```
boolean b = true
while(b) {
        System.out.println("This is true!");
}
```

`for` loop

For loops within the java language have two prolific uses. The common C-like method of defining variables, condition and update. This is represented in the form:

`for(`*variables*`;`*condition*`;`*update*`)`

```
for(int i = 0; i < 10; i++) {
        System.out.println(i);
}
```

The alternative method is an iterator based method which will allow the programmer to express a loop using a binding and collection. This is represented in the following form:

`for(`*binding*`:`*collection*`)`

```
String[] array = { "Hello", "Loops!" }
for(String s : array) {
        System.out.println(s);
}
```

Question 1: Count vowels

You are to write a program that will count the number of vowels that exist in a given string.

```
static int countVowels(String s)
```

```
String s = "astronaut";
int count = countVowels(s);
System.out.println("Number of count: " + count); //4
```

Question 2: Draw a tree

You are tasked with writing a function that will draw a tree to standard output.

```
static void tree()
```

```
    *
   ***
  *****
 *******
   ***
```

Arrays

Arrays are defined as contiguous blocks of memory that contain a number of elements. Memory is allocated based on the size of the type and the number of elements defined. When using primitive typed arrays, all elements are aligned next to each other while reference typed arrays will contain a memory reference. The size of a memory reference is the size of the address space size of the CPU on your computer. This is commonly referred to as the pointer size.

```
int[] array = new int[10]; //By default, initialised to 0
```

Elements of primitive typed arrays will always contain a value. In previous instances of java (<= Java 6), did not gaurantee to be initialise each element to 0.

```
String[] array = new String[10]; //By default, initialised to null
```

Elements of reference typed arrays will by default be initialised to null. By default, each element within the array is a memory reference and does not have any usable data. To set an element within a referenced typed array, you are able to set an element to null or to an object (either directly initialising it with new or through aliasing).

Question 3: Contains

Write a method that will check if an element exists within an array.

```
static boolean contains(int[] a, int element);
```

Example:

```
int[] array = {1, 1, 5, 5, 5, 3, 8, 1};

boolean result = contains(array, 5); //true
```

Question 4: Count

Extend your previous question to return the number of times an element exists in an array.

```
static int count(int[] a, int element);
```

Example:

```
int[] array = {1, 1, 5, 6, 5, 3, 8, 1, 9, 2, 8};

int result = count(array, 1); //3
```

Question 5: Count duplicates

Given an array, count the number of duplicates that exist in a given array.

```
static int countDuplicates(int[] a)
```

Example:

```
int[] dups = {1, 1, 5, 6, 5, 3, 8, 1, 9, 2, 8};

int result = countDuplicates(dups);
//3
```

Question 6: Array Intersection

Given two integer arrays, return an array that will provide the set intersection between array x and array y.

```
static int[] intersection(int[] x, int[] y)
```

Example:

```
int[] x = {1, 2, 5, 6, 0, 8, 7};
int[] y = {3, 1, 2, 6, 10};

int[] result = intersection(x, y);
//{1, 2, 6}, it is not necessary to sort the array
```

Question 7: Ranking athletes

You are given a set of times as a (`float[]`) floating point array. You are tasked with writing an algorithm that will sort the athletes from fastest to slowest and output the time difference between $athlete_{\text{i}}$ and $athlete_{\text{fastest}}$

```
static float[] rank(float[] times)
```

Example output:

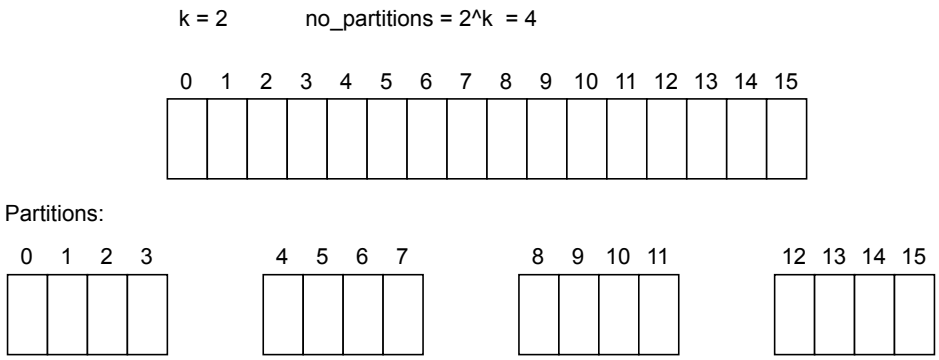
```
Athlete 1: +0.0000
Athlete 2: +1.2340
Athlete 3: +2.1100
Athlete 4: +2.1202
```

Question 8: Assessed Task: Online Task 1

Remember you are required to complete a Online Task within the due date. Go to EdStem for this unit and click on Assessment to find out the task and the due date. This is a marked task.

Extension

Question 9: Binary Partitioning



Given an array of integers and an integer k as an arguement, you are tasked with writing a function that will return an array of partitions. Each partition represents a segment of the array that can be subdivided 2^k times.

```
static int[][] partition(int[] array, int k)
```

Example 1:

```
int[] x = {3, 2, 4, 7};
int[][] result = partition(x, 1);
//{ {3, 2}, {4, 7} }
```

Example 2:

```
int[] x = {3, 2, 4, 7, 8, 9, 2, 3};
int[][] result = partition(x, 2);
//{ {3, 2}, {4, 7}, {8, 9}, {2, 3} }
```