Arrays

An array is a collection of variables of the same type, stored in a contiguous block of memory.

- Arrays are useful for storing and manipulating large amounts of data *efficiently*.
- Arrays have a *fixed* size, which must be specified when the array is created.
- Arrays are indexed starting at 0.

Declaring and Initializing Arrays

```
int[] arrayName;

arrayName = new int[5];
```

Declaring an Array

```
int[] arrayName; // Declare an array variable
```

- The variable `arrayName` is declared as an array of `int` type
- The square brackets `[]` following the type indicate that this variable is an array
- At this point, the array variable has been declared, but it does not have an actual array assigned to it yet
- The memory for the array will be allocated later when we use the `new` operator

Initializing an Array

```
arrayName = new int[5]; // Allocate memory for array
```

- The `new` operator is used to allocate memory for an array.
- It is followed by the type of elements the array will hold and the size of the array in square brackets.
- In this case, we are creating a new array of `int` type that can hold 5 elements.
- The reference of this new array is assigned to the `arrayName` variable.

Declare and initialize arrays

- declare and initialize 2 array variables of different types and different sizes

Accessing and Assigning Array Elements

```
int[] myArray = new int[5];

myArray[1] = 10; // Assign a value to the first element

int value = myArray[1]; // Access the first element
System.out.println(value);
```

- Array elements are accessed by their index, which is a zero-based integer
- The first element is at index 0, the second element at index 1, and so on.
- Elements can be both accessed and assigned values by using the index inside square brackets `[]`.
- ***Predict: what is the output?***

Accessing and Assigning Array Elements

```
int[] myArray = new int[5];

myArray[1] = 10; // Assign a value to the first element

int value = myArray[1]; // Access the first element
System.out.println(value);
```

- **Answer**: the output is `10`
- *Question: Which element of the array has been assigned to?*

Accessing and Assigning Array Elements

```
int[] myArray = new int[5];

myArray[1] = 10; // Assign a value to the first element

int value = myArray[1]; // Access the first element
System.out.println(value);
```

- Question: Which element of the array has been assigned to?
- **Answer:** The first element (element 0) has been assigned to

What happens if you access a slot of an array before assigning to that slot?

What happens if you access a slot of an array before assigning to that slot?

- **Answer**: when you make an array every slot is filled with a default value!
    - the default value depends on the type of the array

Array slots and default values

- ***Question:*** What is the default value for arrays of type `int`?
- ***Question:*** What is the default value for arrays of type `double`?
- ***Question:*** What is the default value for arrays of type `boolean`?
- ***Question:*** What is the default value for arrays of type `String`?
- ***Question:*** What is the default value for arrays which store reference types?

Array slots and default values

- Question: What is the default value for arrays of type `int`?
  - *Answer:* ***`0`***
- Question: What is the default value for arrays of type `double`?
  - *Answer:* ***`0.0`***
- Question: What is the default value for arrays of type `boolean`?
  - *Answer:* ***`false`***
- Question: What is the default value for arrays of type `String`?
  - *Answer:* ***`null`*** (remember `String` is a reference type!)
- Question: What is the default value for arrays which store reference types?
  - *Answer:* ***`null`***

What happens if you try to access an index that doesn't exist?

What happens if you try to access an index that doesn't exist?

- ***Answer:*** `ArrayIndexOutOfBoundsException`
- Remember that unless you do fancy stuff, any exception crashes the program!!