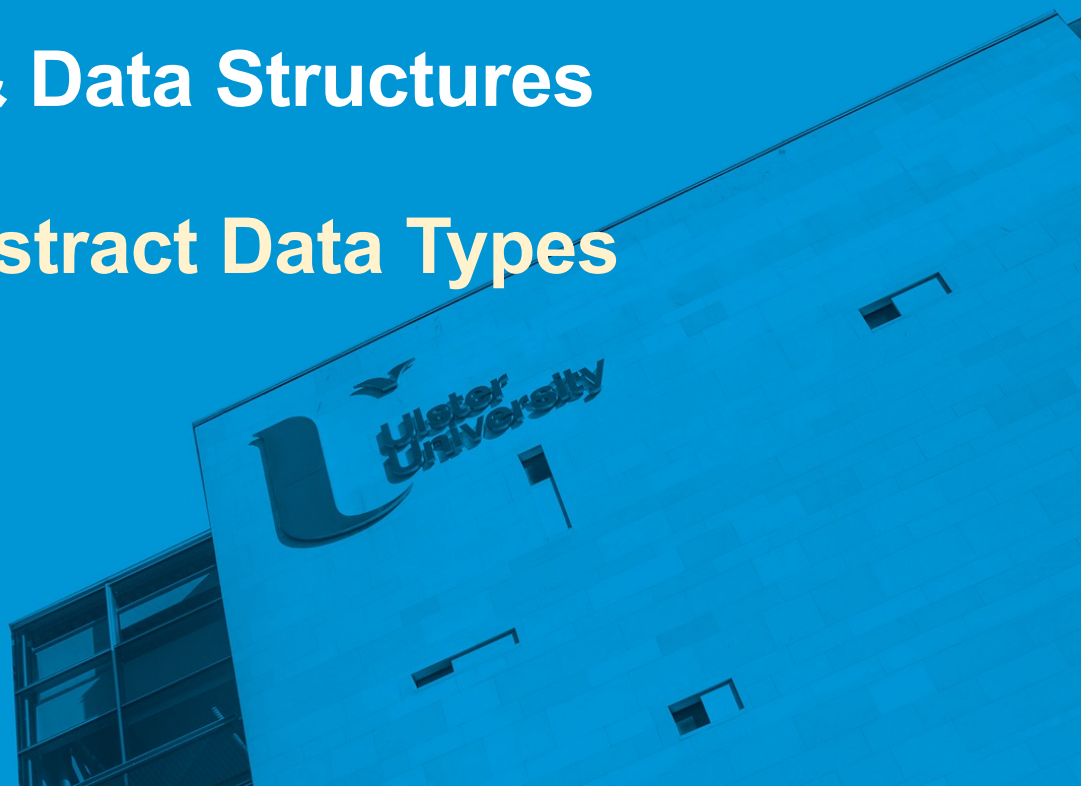
Ulster University

COM498 Algorithms & Data Structures

2.1 Introduction to Abstract Data Types



Abstract Data Types

- A data type is set of values and operations on those values (defined within a specific programming language)

- Primitive types:
  - values map to machine representations
  - operations map to machine instructions

- We want to write programs that process other types of data!

type	set of values	examples of values	examples of operations
char	characters	'A' '@'	compare
String	sequences of characters	"Hello World"	concatenate
int	integers	17 12345	add, subtract, multiply, divide
double	floating-point numbers	3.1415 6.022e23	add, subtract, multiply, divide
boolean	truth values	true false	and, or, not

- An abstract data type is a data type whose representation is hidden from the client

Abstract Data Types

Ulster University

- A more specific definition:

An **Abstract Data Type** (ADT) is a specification for a group of values and the operations on those values that is defined conceptually and independently of any programming language.

(a data structure is an implementation of an ADT within a programming language)

Abstract Data Types

- You may have already seen (and been been using) the implementation of an ADT when manipulating string values

- Java's String data type is an ADT that allows programs to manipulate strings
- A String is a sequence of Unicode characters
- The API for String provides an interface for the ADT

public class String	
String(String s)	create a string with the same value
int length()	string length
char charAt(int i)	ith character
String substring(int i, int j)	ith through (j-1)st characters
boolean contains(String sub)	does string contain sub?
boolean startsWith(String pre)	does string start with pre?
boolean endsWith(String post)	does string end with post?
int indexOf(String p)	index of 1st occurrence of p
int indexOf(String p, int i)	index of 1st occurrence of p after i
String concat(String t)	this string with t appended
int compareTo(String t)	string comparison
String replaceAll(String a,String b)	result of changing as to bs
String[] split(String delim)	strings between occurrences of delim
boolean equals(String t)	is this string's value the same as t's

- Remember we can use an ADT without knowing the underlying implementation details

Abstract Data Types

Ulster University

- When using data types, you need to know:
  - Its name (capitalized in Java)
  - How to construct new objects (instantiation)
  - How to apply operations on to a given object (invoke methods)

- To construct a new object:
  - Use keyword new to invoke a constructor
  - Use data type name to specify the type of the object

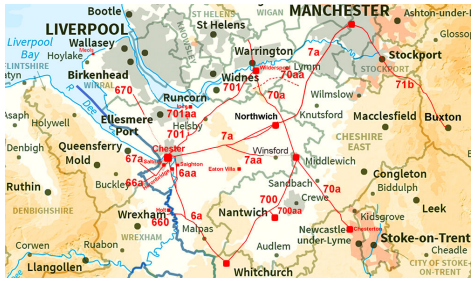
- To apply an operation:
  - Use object name to specify which object
  - Use the dot operator to indicate an operation is to be applied

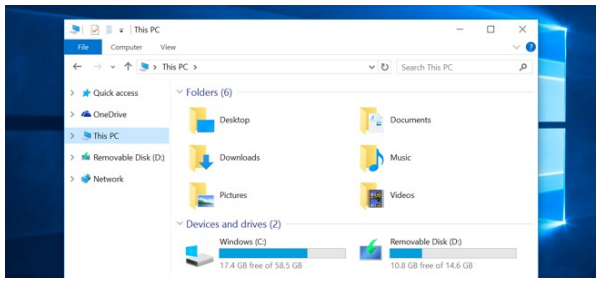
```
String str;
str = new String("Hello world");
str = str.substring(0, 5);
System.out.println(str);
```

Using an ADT is like using a vending machine

- Can perform only tasks machine's interface presents
- You must understand these tasks
- Cannot access the inside of the machine
- You can use the machine even though you do not know what happens inside
- Usable even with new insides

Examples of Data Organization





- Using an ADT allows us to generalize the idea of grouping objects as collections

Scenario

Ulster University

- Define an Abstract Data Type called `Animal` that stores the following on various kinds of animals
  - The kind of food the animal eats (e.g. “meat”, “vegetation”, “fruit” etc.)
  - The average life expectancy of the animal (e.g. 1, 10, 20, 100 – where the value is in years)

- Create a new Java project called Animal and implement your ADT as a new class `Animal` in a file ***Animal.java***
  - Provide 4 versions of the constructor to allow instances of Animal objects to be created (i) when no parameters are provided, (ii) when only a kind of food is provided, (iii) when only a life expectancy is provided, and (iv) when both values are provided
  - Write a `toString()` method for the `Animal` class that outputs a sentence describing the animal in terms of its food type and life expectancy
  - Provide a `main()` method that creates 4 animals and prints their descriptions

Challenge

Lions and Animals

- In your `Animal` project, implement a sub-class of `Animal` called `Lion` and give it the instance variables and methods as described in the **Lions And Animals Challenge Specification** available on Blackboard.

- Create the test application class `TestAnimal` as directed, and implement the following
  - Create two Lion objects called `myLion1` and `myLion2`. Initialise the properties required with values of your own choice
  - Make a call to an appropriate method to set the age of `MyLion1` to 3.
  - Print out the details of `myLion1` using the `toString()` method and the number of Lions created using the `numberOfLions()` method.