COMP08034 Structures & Algorithms – Coursework

Session 2020-21

This coursework can be carried out either individually or in pairs. Submission should be via Moodle. You are to implement a dynamic data structure for a small pet supplies company to maintain details of their products.

You are advised to develop the program one step at a time and keep a copy of the version for each step which should then be included in your submission. For steps 1 & 2, you should use **different** dynamic structures. Step 3 requires you to join the two structures together. In addition to the marks at each step, there are 20 marks for documentation, programming style and a report. Your report should cover how you tested your system and discuss any difficulties you encountered including any parts of your solution that are not working correctly (see marking scheme). There are 80 marks for this coursework and it is worth 40% of the marks for the module.

A small pet supplies company would like a system to hold a list of products categorised by type of pet suitable for.

The data to be stored for each type of pet is:

Pet Type / Name e.g. Cat, Dog, Fish

There should also be a link to the product Information for each type of pet. The data to be stored for each product is:

Product Name e.g. Lead

Product Code e.g. P999

Quantity in stock e.g. 100

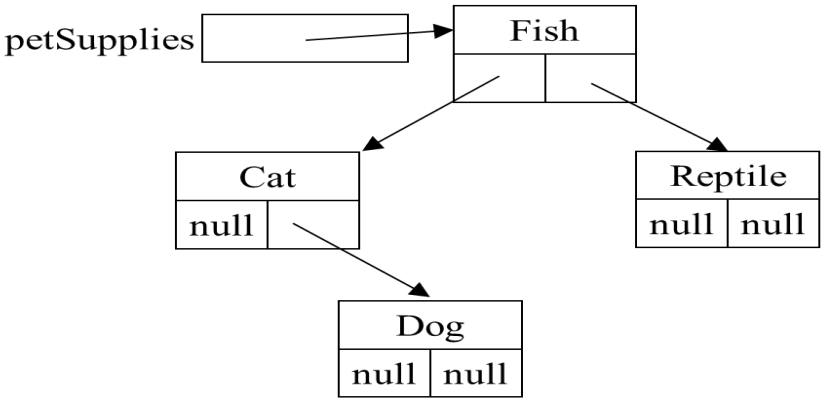
The features required by the company include:

- Add details of a new pet type
- Add details of a new product for a specified pet type
- Find if the company stocks products for a particular type of pet
- Display all the types of pets that the Company provides products for
- Display all the products stocked for a specific pet type
- Display all the product details for all the pets
- Remove a particular product for a specified pet (eg discontinued ítem)
- Remove a pet type from the system

Step 1: Functionality is worth 15 marks

- select a suitable dynamic data structure (e.g. BST) to hold details of the categories of pets the company stocks products for and create a **Pet** class with suitable attributes to represent each pet type (at this stage a link to each product for that type of pet is not required although you may wish to include it).
- write a **PetTest** class that offers a simple text-based menu to:
    - **add** a new pet type;
    - **find** if the company provides products for a particular pet type;
    - **display** a specific pet type;
    - **remove** a specific pet type;
    - **display** the details of all the pet types the company provides products for in alphabetical order.

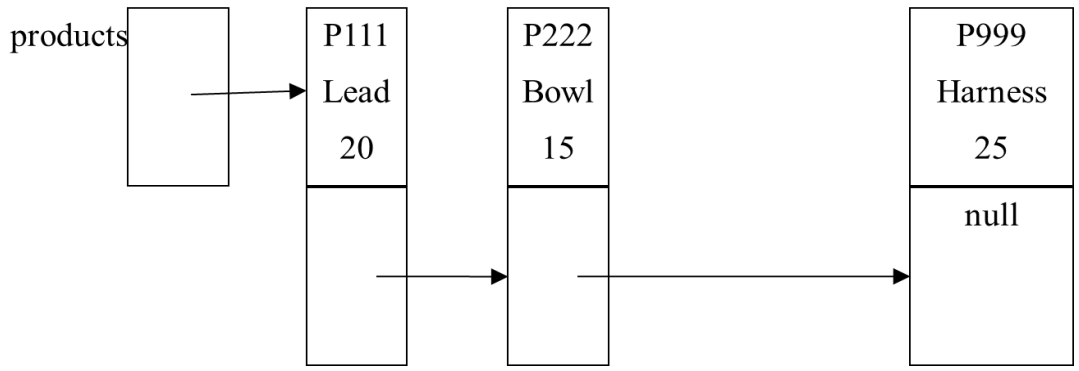
A possible structure is:



Step 2: Functionality is worth 15 marks

- select a suitable dynamic data structure (e.g. Linked List) to hold product details and create a **Product** class with suitable attributes for each type of product. The structure chosen should be different to that used in step 1.
- write an **addProduct** method that interacts with the user to add the pet product details;
- write a **findProduct** method which searches for a particular product and displays the details if found;
- write a **removeProduct** method to remove a specific product from the system if they are no longer stocked;
- write a **displayProducts** method which displays all the products;

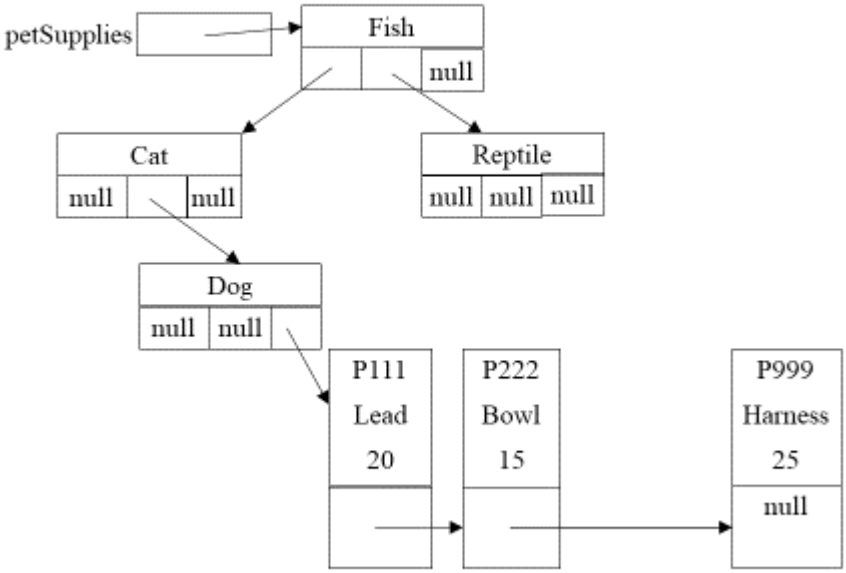
A possible structure is:



Step 3: Functionality is worth 30 marks

- combine your structures/ programs from steps 1 & 2 to create a program which holds a list of products for each pet type.
- Your menu should now provide options to:
    - Add details of a new pet type;
    - Add details of a new product for a specified pet type;
    - Find if the company stocks products for a particular type of pet;
    - Display all the types of pets that the company provides products for;
    - Display all the products stocked for a specific pet type;
    - Display all the product details for all the pets;
    - Remove a particular product for a specified pet (eg discontinued ítem);
    - Remove a pet type from the system.

A possible structure is:



Submission

Due Date: Sunday 9th May 2021 @ 23:59

Upload your work via Moodle

- create a folder based on your banner id e.g. *B00123456.* If the coursework is done in pairs, then include both Banner ids.
- copy **ALL** the projects for the different versions of the program into this folder as well as your report
- create a compressed zip version of this folder
    - in Windows Explorer, select the folder with the right button then select Send To then Compressed (zip) Folder
- select the Coursework Submission link in the Coursework tab on Moodle and follow the instructions to upload the zip file

Marking Scheme

		Max	Actual	
Step 1	**Half marks if the structure is not a tree.**			
	add pets to chosen dynamic structure	3		
	checks if pet type already exists	2		
	find a particular pet type & display info	3		
	suitable message if pet not found	1		
	remove pet from structure	2		
	display all pets in alphabetical order	2		
	suitable message if no pets in the system	1		
	message if invalid menu option entered	1		0 / 15
Step 2				
	add product info to chosen dynamic structure	3		
	checks if product already exists	2		
	suitable message if product already in the system	1		
	find a particular product & display info	2		
	suitable message if product not found	1		
	remove product from structure	3		
	display all product details	2		
	suitable message if no products in the system to display	1		0 / 15
Step 3	**No marks for this section if not using a combined structure**			
	add pets to the **new** structure	2		
	message if pet already exists	1		
	add product details for a specific pet type	3		
	when select a pet type			
	detects no match for pet	1		
	suitable message displayed	1		
	only ask for product information if pet type exists	1		
	Find if the company stocks products for a particular type of pet and display info	2		
	suitable message if not found	1		
	deletes a pet type from the system	2		
	displays pet removed message	1		
	deletes a product type for a specified pet type	2		
	displays product removed message	1		

	display					
	display all the types of pets that the company stocks products for	2				
	displays product details for a specific type of pet	2				
	display all the product details for all the pets in alphabetical order of pet	3				
	Add JUnit classes to test the solution	5		0	/	30
Style & report						
	code layout/indentation	1				
	meaningful names	1				
	appropriate use of public/private	1				
	appropriate use of static identifier	1				
	appropriate override of Object methods	1				
	use of *this* to identify object components	1				
	adherence to structured programming	1				
	Source code in Github repository	1				
	Evidence of use of the Github repository	1				
	Exception handling	3				
	Report including testing	8		0	/	20
			Total	0	/	80

NOTE - No marks will be awarded if the source code does not compile