



eduresourceza.com



BEAULIEU COLLEGE
INFORMATION TECHNOLOGY
GRADE 12
PRACTICAL EXAM
JULY 2009

TIME: 3 HOURS
EXAMINER: MS S. CRAIG

MARKS: 120
MODERATOR: MR P. NORTON

INSTRUCTIONS:

1. Read through the paper and follow all instructions CAREFULLY.
2. Save all your work in the exam folder provided.
3. Use the templates given to you in order to save time.
4. Ensure that your name is on every class that you code.
5. Good luck, me hearties!! Arrr!



SCENARIO

According to the “Period Table of Awesomeness”, shown below as copied from the website *DapperStache.com*, Pirates are awesome number nine for being amongst the most awesome things in existence. Chuck Norris is awesome 11¹.

This has nothing to do with your practical exam, except that it gives us an awesome topic!

PIRATES!

(Awesome 9)

The Periodic Table of Awesomeness

Group 18: The Noble Races

1 Bn Bacon												13 Mt Mt. Dew	14 Br Beer	15 C Chocolate	16 Ch Cheese	17 P Pirates	18 Sn Sniper										
3 Bt Batman	4 Ex Explosion												13 Bb Boobs	14 Bf Beef Jerky	15 Cf Coffee	16 Sg Sausage	17 Zb Zombies	18 As Assassin									
11 Chuck Norris	12 Gr Grenades	3 L Laser	4 Jf Jet Fighter	5 Th Throwing Stars	6 Mg Minigun	7 Sb Sonic Boom	8 Kf Kung Fu	9 Jp Jet Pack	10 Hc Helicopter	11 Mw Microwave (a.c.b.)	12 Ls Light Speed	31 S Space	32 Tb Taco Bell	33 Kl Kool Aid	34 Rm Ramen	35 Dw Dwarves	36 V Vampire										
19 Ba Mr. T	20 Bm Bombs	39 Sw Sword	40 Tk Tank	41 Nc Nunchucks	42 Dg Daggers	43 Tr Trebuchets	44 Ob Ob Light Saber	45 Mj Moon Jumps	46 Mc Mecha	47 I Internet	48 Hs Hyper-space	49 Lt Lightning	50 F Fire	51 Lq Liquor	52 Do Doritos	53 R Robots	54 Wz Wizard										
37 Mt Samuel L. Jackson	38 Rt Rockets	72 Mo Monster Truck	73 Me Motorcycle	74 Ax Battle Axe	75 Cp Catapult	76 Ry Ray guns	77 Vg Video Games	78 T Teleport	79 Co Computers	80 Rw Time Travel	81 Tn Tornado	82 Bh Black Holes	83 St Storms	84 Rb Ribs	85 An Aliens	86 Jd Jedi											
55 Sh William Shatner	56 Fw Fireworks	104 Bs Battleship	105	106	107 Ft Fortress	108	109	110	111 Lg Legos	112	113 Sx Sex	114 M Metal	115 Mg Magnets	116 My Money (a.c.)	117 Bo Bounty Hunters	118 Gh Ghost											
87 Wn Christopher Walken	88 Pm Proximity Mines												139 D Dinosaur	140 Sk Shark	141 Dr Dragon	142 Gs Giant Squid	143 Pg Penguin	144 Ea Eagle	145 Ct Cheetah	146 Cr Crocodile	147 Wf Wolf	148 B Bear	149 Cb Cobra	150 F Falcon	151 Tr Trilobite	152 Wv Were-wolves	153 Mu Mutants
The Dapperstache																											
																											
89 Ds Mustache 90 Mk Mohawk 91 Yy Ying Yang 92 Sl Skulls 93 Sc Scars 94 Ta Tattoos 95 Ml Mullet 96 Hd Hoodies 97 Su Sunglasses 98 So Soul 99 Hg Holograms (T.A.) 100 101 102 103																											

<http://www.dapperstache.com/index.php?contenttype=ptoa>

¹ Because he's Chuck Norris, his symbol is "Chuck Norris". No symbol can represent Chuck Norris' awesomeness

SECTION A: DATABASE AND SQL

You are given the following tables in the database "Pirates.mdb".

tblShips		
ShipID	ShipName	PirateID
1	Albatross	1
2	Osprey	2
3	Black Pearl	3
4	Locker	5
5	Shivering Timb	4
6	Argonaut	6
7	Team Craig	7
8	The Samoosea	8

tblPirates							
PirateID	Surname	FirstName	Gender	Weight	Height	MedID	KgLost
1	Bart	Big	M	105	1.93	1803	12.4
2	Gertie	Green Eyed	F	80.2	1.3		31.2
3	Sparrow	Jack	M	74.3	1.64	1204	12.4
4	Boris	One Handed	M	104.8	1.58	1302	25.4
5	Jones	Davy	M	94	1.66		28.6
6	Craig	Cap'n	F	55	1.68	1803	-5.9
7	Hook	James	M	85.1	1.62	1907	13.6
8	The Terrible	Yassi	F	58	1.84		0.6

tblMedCondition	
MedID	Description
1204	High Blood Pressure
1302	Parrot Flu
1407	Hangman's Neck
1502	Amputee
1604	Acrophobia
1701	Back Problems
1803	Severe Scarring
1907	Yardarm Jaundice

You are given the class **PirateDB** which contains code necessary to connect to the database and which provides a basic menu structure. You may adjust this menu structure if you wish, but no extra marks will be awarded for this. Indeed, you may end up penalising yourself.

Open the program named **PirateDB**

1.1 Menu Option A:

Display the names, gender and weights of the pirates stored in the database, sorted from shortest to tallest. (4)

1.2 Menu Option B:

Ask the user to input a medical code and then display corresponding medical condition. Below the condition, list all the pirates who happen to have that condition. (8)

1.3 Menu Option C:

Display the start weight of each pirate, then use the KgLost field to calculate and display the current weight of each pirate. (5)

1.4 Menu Option D:

Ask the user to input a gender abbreviation (F, M) and then display all the pirate ships whose captains are of that gender. (6)

1.5 Menu Option E:

Write a code to add a new pirate and a new ship to the database. The relevant fields in both tables (tblPirate, tblShips) must be completed. (7)

[30]

SECTION B:

OBJECT ORIENTED PROGRAMMING

Your program must allow you to enter in details from a textfile (supplied: Pirates.txt). These details must be read into an array and manipulated in various ways to create a fun and exciting game.

PLEASE START A NEW NETBEANS PROJECT. This program does not rely or require any of the code that you created in Section A.

The following tasks must be performed:

- A: Display pirate data
- B: Search for a pirate
- C: Go into battle
- D: Sort total points (descending)
- E: Display all Pirates with a medical condition
- F: Display ship statistics of a Pirate
- Q: Quit

Your application will consist of the following classes:

Pirate (provided, but you must change and add on as needed)

PirateMedCondition

PirateArray

PirateApp (provided, but you must change and add on as needed)

QUESTION 1:

- 1 Open the class called **Pirate**. Define appropriately named and typed class variables for the following:
 - a. The pirate's name
 - b. The ship strength
 - c. The number of crew members
 - d. The number of cannons on board
 - e. The number of bottles of rum

3.

- 2 Write a parameterised constructor that will initialise the variables. 3.
- 3 Complete the accessor methods that have been supplied in the class template. 3.
- 4 Code a method called **totPoints** that will calculate the total hit points of the pirate object using the following formula:

HP = size of the ship + (number of crew x 2) + (number of cannons x 3) – (the amount of rum x 2) + a luck factor which should be between 0 and 10. 4.
- 5 Code a **toString** method that returns the name of the pirate as well as the total hit points of that pirate. 3.

[16]

QUESTION 2:

1. Open the class called **PirateMedCondition** which inherits from **Pirate**. *(There are no properties in this class. Below, however, we will override a method to change how the points are calculated for these Pirates.)* 3.
2. Write a parameterised constructor for this class. It must call the super class constructor to initialise the parent class properties. 3.
3. Code a **totPoints** method which will override the method in the super class. This must return the total hit points of the **PirateMedCondition** object by multiplying the number of points, as calculated by the method in the parent class, by 1.15 to give pirates with a medical condition a small handicap advantage. 4.
4. Code a **toString** method which will call the **toString** method of the super and then concatenate the number of points that are added on for the handicap (0.15 x the value of the points of the parent class). 3.

[13]

QUESTION 4:

Please read through this question carefully before you start to code it. It is important that you understand what is hoping to be achieved.

1. Create a class called **PirateArray** in which you declare a **Pirate** array of 100 elements, as well as a variable to keep track of how many **Pirate** objects are in the array. 4.
2. Create a constructor method which will read the data from the text file **Pirates.txt** and assign the values to the **Pirate** array. Your code must take into consideration that some elements in the array will be **Pirate** objects, others **PirateMedCondition** objects. **PirateMedCondition** objects are indicated with a value of "true" in the last field. 10.
3. Code a **toString** method which will display the name, total points, medical handicap of each item (where applicable) in the array. Ensure that suitable headings are displayed. (this will be used later by menu option A). 4.
4. Code a method called **doBattle** which will allow the user to choose 2 pirates in the array who will do battle together (do this by asking the user to indicate their positions in the array and the pass these two numbers through to the method as parameters). The winner is calculated as follows: the pirate has the total hit points as calculated in the **Pirate** class, and then has the added bonus points if he has a medical condition (because of our use of inheritance, this is automatically taken into consideration). The pirate with the higher points wins. Show a suitable output statement to give the result of the battle. (this will be used later in menu option C) 10.
5. Code a typed method called **search** that takes in a String as a parameter and returns a **Pirate** object. This will allow for the user to enter the name of a pirate and have all the relevant details of that pirate displayed as output. (this will be used later by menu option B) 5.
6. Code a method called **findPirateMedCondition** that will print out all instances of **PirateMedCondition** objects in the array. (this will be used later by menu option E) 5.

7. Code a method called **sort** which will print out the name of the **Pirate** and his stats: the size of ship, number of crew, number of cannon and the bottles of rum. Rank the output descending according to the number of points. (this will be used later in menu option D) 10.
8. Code a method called **displayShipStatistics** which will allow the user to enter the name of a **Pirate** as a parameter and then have the details of his ship displayed on the screen. (this will be used later in menu option F). 6.

Bonus: Questions 4, 5 and 8 all search for pirates based on their names, although the output from the methods is different. Can you create a method that will do the searching for you? 5.

[54 + 5]

QUESTION 5:

1. Use the template called **PirateApp** to make the appropriate method calls in the switch statements in order to run the application according to the given menu. 7.
[7]