

**CG1103 - DATA STRUCTURES AND ALGORITHMS I**

Group Lab 3

Lab Day: Friday Lab Time: 1100 - 1300

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Matric Number | Email | Contribution to  Lab 4 |
| Team Leader | Yong Jia Hong | A0087736N | a0087736@nus.edu.sg | Implement the jobs algorithm and checking of inputs |
| Member 1 | Yang Mo | A0091836X | a0091836@nus.edu.sg | Improvements to the data structure, implementing of the jobs algorithm and a logging in function |
| Member 2 | Dinesh Thangavel | A0088592M | a0088592@nus.edu.sg | Improvements to the data structure and beautifying the program |

**To run batchjobs**: Firstly, choose the type of container to use for storage of data. After that choose option 2 ("Manager") and key in password (12345) to authenticate. Secondly, choose option 8 to load data from ("batchdata.txt"), after that choose option 11 to load jobs to be done from ("batchjobs.txt"). At this point in time, the job processing is not done yet, it is only done after user keys in option 12 and the changes to the database will be updated and a file ("log.txt") is generated to record any errors.

**Explanation**: Since the jobs of the first user in the file ("batchjobs.txt") is to be executed last and vice versa, jobs of the particular user is firstly stored in a queue and the queue is stored in a stack and then stored in another queue again. This is because, before the user opts to process the jobs, he could choose to load another jobs file and to ensure that the system process the jobs in the correct order, which is jobs that is loaded first is executed first.

Changes in Glab4

1. Main Function:

In main function, we added a choice statement to identify the user’s status before any products information is loaded (Whether the user is a normal user or a Manager). If it is a manager, then a string of password is needed for verification. When the program is launched for the first time, the password for manager is initialized to be a simple string of numbers(12345), but the manager can update the password to a more complicated and secure one if needed. Moreover, the password will be remembered when next time the system is launched. Also, this password cannot be hacked by any other user as it is encrypted using shift transformation.

Also, logging into the system as a normal user will not be required to provide a password but when a normal user wants to log in the manager account, the system will automatically exit when the maximum number of attempts is exceeded (which we set to be 5 times).

The normal user will only have access to a function-limited menu with much fewer choices than the manager’s.

1. API:

1)Changing return types of several functions (add delete eta) as well as adding num\_to\_sell and num\_to\_add as new parameters for functions

API\_SellProduct & API\_RestockProduct to adapt to the needs of Job Processing class.

2)Added Private variables:

a)Jobdata which is a processJobs class object;

b)A product type pointer ;

3)New functions:

a)API\_loadJobData and API\_proceedjobs (used to process job data);

b)API\_displayprofeatures(product\*) to call the combined display function in productdatabasebase instead of using respective displayfunctions in three databases;

c) API\_checkInt(string toCheck) and API\_checkDouble(string toCheck) to check that the user inputs the correct characters

1. UI
2. Added private variable ---password;
3. Added functions keeppassword() and loadpassword() to track the password;
4. Added function UI\_display\_pro\_features(product\*) to use UI to do the product info displaying work;
5. Added function checkpassword to do the verification in main;
6. Added function getnormaluserpage() to print a limited function menu for normal users;
7. Added functions setpassword as well as changepassword to initialize and reset the password;
8. The UI has been made attractive by using colours and this application has been designed for a user using Windows operating system.

1. New Class---processJobs

1)use queue to store each user’s jobs and then use a stack to store the users;

2)use a structure to store the necessary information;

3)a void proceedjobs function in which:

dojobs function is called repeatedly for each job of each user; this function has a bool return type, which means that if current job accessed can be done then return true, else return false and the error job’s information will be immediately written into the log.txt file;

1. Productdatabasebase and 3 databases:

1)bellow functions:

void listallproducts();

void searchbyName(string Name);

void searchbyManu(string Manufacturer);

void searchbyBarcode(long unsigned int Barcode\_num);

void searchbyCategory(string Category);

void TOPXsellingproduct(int x);

void Bestsellingcategory(string Category);

void Bestsellingmanufacturer();

have been combined into common functions in productdatabasebase (Abstract class) instead of being overloaded respectively in each of 3 databases;

2) Added function Dispose(long unsigned int Barcode\_num, string currentdate) to proceed the disposing jobs in Glab4;

3) Added pure virtual function product\* traverse (int) in order to traverse the product in different databases to return a common pointer pointing to products, which is necessary for combining the functions in 5.1 mentioned above; this function is called repeated in the combined functions;