APRIL 6, 2016

ADVANCED PROCEDURAL PROGRAMMING PROJECT

CREATE A DATABASE TO STORE SURVEYS FOR CUSTOM-SURVEYS LTD

KEITH WILLIAMS

G00324844

TABLE OF CONTENTS

ABOUT

This program was written in **Visual Studio Community 2013**.

This program will create a database which will store survey details and generate statistics about those surveys.

This project requires a file called users.txt, which holds user credentials, i.e. a username and password, for at least one user. If this file does not exist the user will not be able to login to this system. I have supplied a sample users file in my submittion.

If you require any more information you can email me at G00324844@gmit.ie.

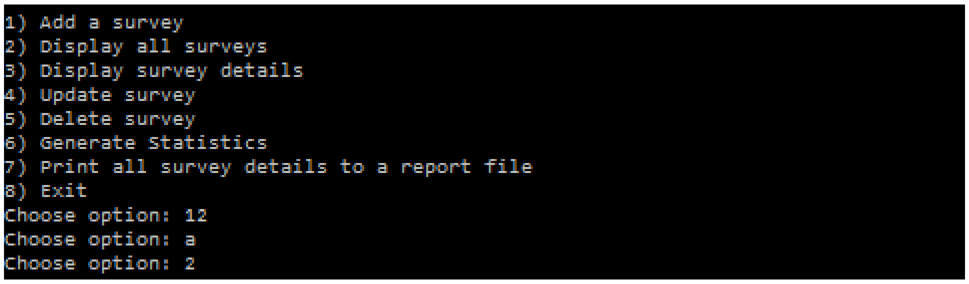
FUNCTIONS

Below is a description of how each major function in this project works followed by a screen shot of the output from that function, if there is one.

MAIN FUNCTION

1. First call the login function. If the user logs in successfully continue. If not, exit the program.
2. Call a function to initialize the linked list of surveys.
3. Display the menu with eight options to the user, including an option to terminate the program.
4. If the user chooses an invalid option it will prompt them to enter another value.
5. When the user chooses to exit the menu update the survey file.

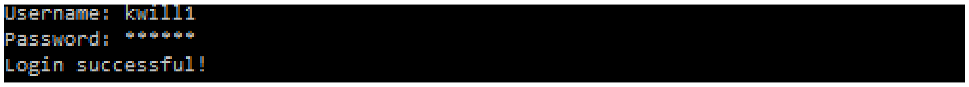
Demonstrating the main menu and the error handling associated with it.


```
1) Add a survey
2) Display all surveys
3) Display survey details
4) Update survey
5) Delete survey
6) Generate Statistics
7) Print all survey details to a report file
8) Exit
Choose option: 12
Choose option: a
Choose option: 2
```

LOGIN FUNCTION

1. This function returns true if the login is successful and false if it is not.
2. All user credentials are stored in a file called “users.txt”. This file is formatted in a particular way so it can be read in correctly. The username and passwords are stored on separate lines so that they can be read using fgets function. This means that usernames and passwords can contain spaces. This function assumes the username and password are 6 characters long.
3. Attempt to open the file in read mode.
4. If it cannot be opened print an error message and return false, meaning the user did not login successfully.
5. Otherwise, read through the file until the end is reached.
6. On each iteration create a new user struct and store the username and password in that struct. Add this struct to the start of the user linked list. This means the users will be listed in reverse order, but order does not matter in the user linked list.
7. Close the file.

8. Get a username and password from the user. The password is inputted in another function which masks the input.
9. Loop through the structs in the linked list and check if the users' credentials match any credentials stored in the linked list.
10. Repeat steps 9 and 10 until there is a match or the user attempts to log in three times
11. Free the allocated memory for all elements in the linked list.
12. If there was a match the login was successful and return true, else the login failed and return false.



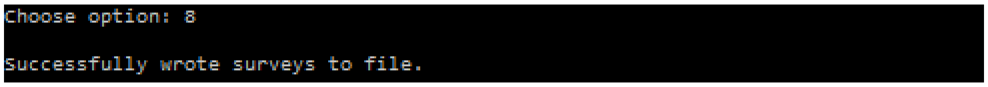
```
Username: kwill1
Password: ******
Login successful!
```

INITIALIZE LINKED LIST FUNCTION

1. All surveys are stored in a file called survey.txt
2. The file is opened in read mode.
3. If there is an error opening the file print an error message.
4. Otherwise, loop through the file survey by survey.
5. On each iteration allocate memory for a new element struct, store details read from the file in the struct and add it to the linked list.
6. Close the file.

UPDATE SURVEY FILE

1. Save the surveys to the survey.txt file.
2. Try to open the file in write mode.
3. If an error occurs print an error message.
4. Otherwise, loop through the linked list and print the data for each survey in a particular format so that it can be read in again by the initializeLinkedList function.
5. Close the file.



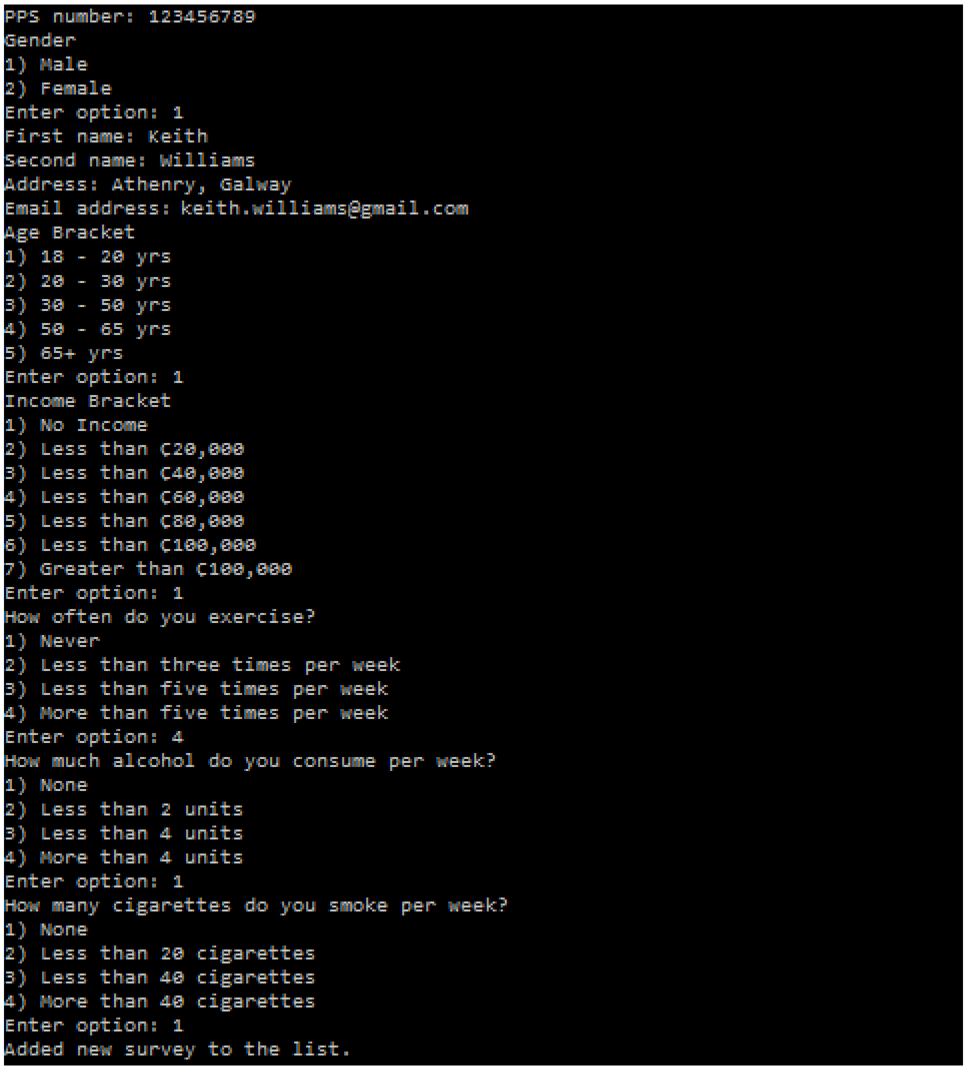
```
Choose option: 8

Successfully wrote surveys to file.
```

ADD SURVEY

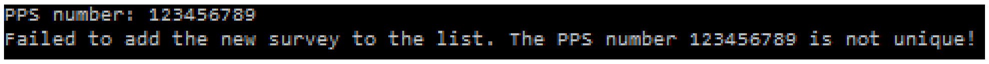
1. Get the PPS number for the new survey.
2. Check if that PPS number already exists in the linked list, using the searchByPPS function.
3. If it does print an error message.
4. Otherwise, allocate memory for a new element in the linked list
5. Allow the user to enter the data for the new survey, excluding the PPS number as they already entered that.
6. Add the new element to the linked list using a separate add function.

Successfully creating a new survey and adding it to the linked list.



```
PPS number: 123456789
Gender
1) Male
2) Female
Enter option: 1
First name: Keith
Second name: Williams
Address: Athenry, Galway
Email address: keith.williams@gmail.com
Age Bracket
1) 18 - 20 yrs
2) 20 - 30 yrs
3) 30 - 50 yrs
4) 50 - 65 yrs
5) 65+ yrs
Enter option: 1
Income Bracket
1) No Income
2) Less than Ç20,000
3) Less than Ç40,000
4) Less than Ç60,000
5) Less than Ç80,000
6) Less than Ç100,000
7) Greater than Ç100,000
Enter option: 1
How often do you exercise?
1) Never
2) Less than three times per week
3) Less than five times per week
4) More than five times per week
Enter option: 4
How much alcohol do you consume per week?
1) None
2) Less than 2 units
3) Less than 4 units
4) More than 4 units
Enter option: 1
How many cigarettes do you smoke per week?
1) None
2) Less than 20 cigarettes
3) Less than 40 cigarettes
4) More than 40 cigarettes
Enter option: 1
Added new survey to the list.
```

Example of error handling when trying to add a duplicate survey (A survey with the same PPS)

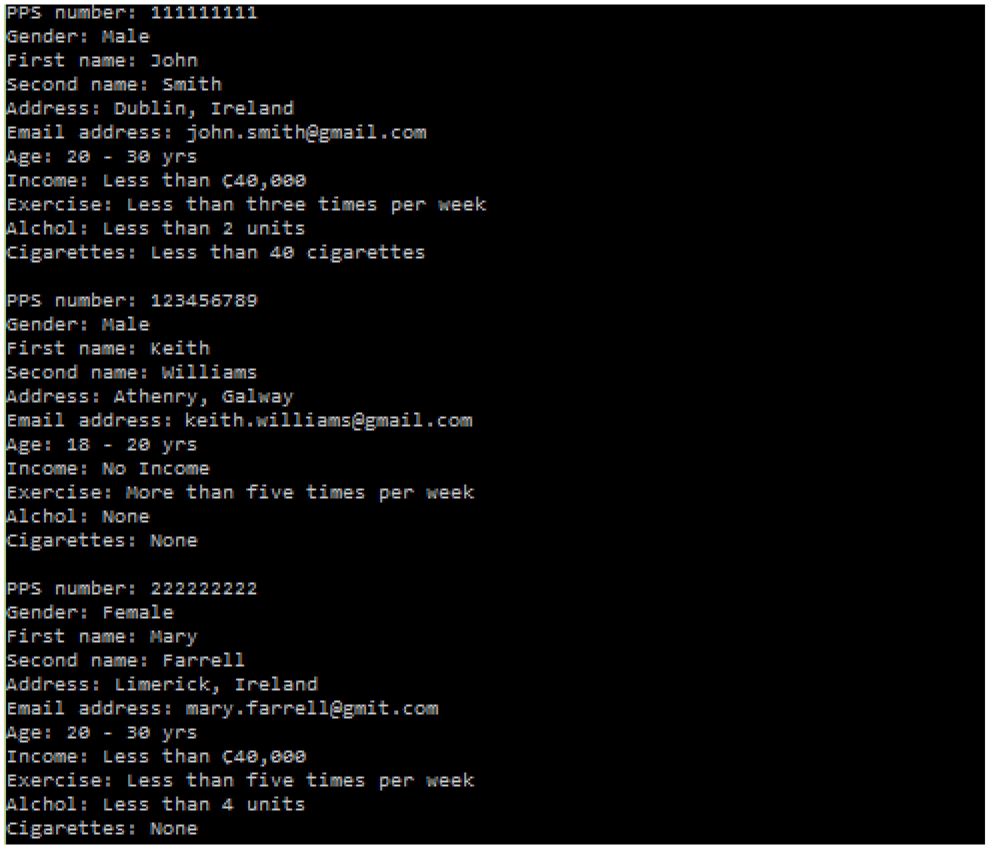


```
PPS number: 123456789
Failed to add the new survey to the list. The PPS number 123456789 is not unique!
```

DISPLAY ALL SURVEYS FUNCTION

1. This function simply loops through all the elements in the linked list add calls the displaySurveyDetails.
2. This function takes two parameters. They are the current element in the linked list and a file pointer which it will print the details to, which in this case is stdout which prints to the console. I chose to pass in a file pointer to avoid writing two similar methods with one using the printf function and the other using the fprintf function.

Outputting all surveys to the console. Notice that they are listed in ascending order based on PPS numbers.



```
PPS number: 111111111
Gender: Male
First name: John
Second name: Smith
Address: Dublin, Ireland
Email address: john.smith@gmail.com
Age: 20 - 30 yrs
Income: Less than €40,000
Exercise: Less than three times per week
Alchol: Less than 2 units
Cigarettes: Less than 40 cigarettes

PPS number: 123456789
Gender: Male
First name: Keith
Second name: Williams
Address: Athenry, Galway
Email address: keith.williams@gmail.com
Age: 18 - 20 yrs
Income: No Income
Exercise: More than five times per week
Alchol: None
Cigarettes: None

PPS number: 222222222
Gender: Female
First name: Mary
Second name: Farrell
Address: Limerick, Ireland
Email address: mary.farrell@gmit.com
Age: 20 - 30 yrs
Income: Less than €40,000
Exercise: Less than five times per week
Alchol: Less than 4 units
Cigarettes: None
```

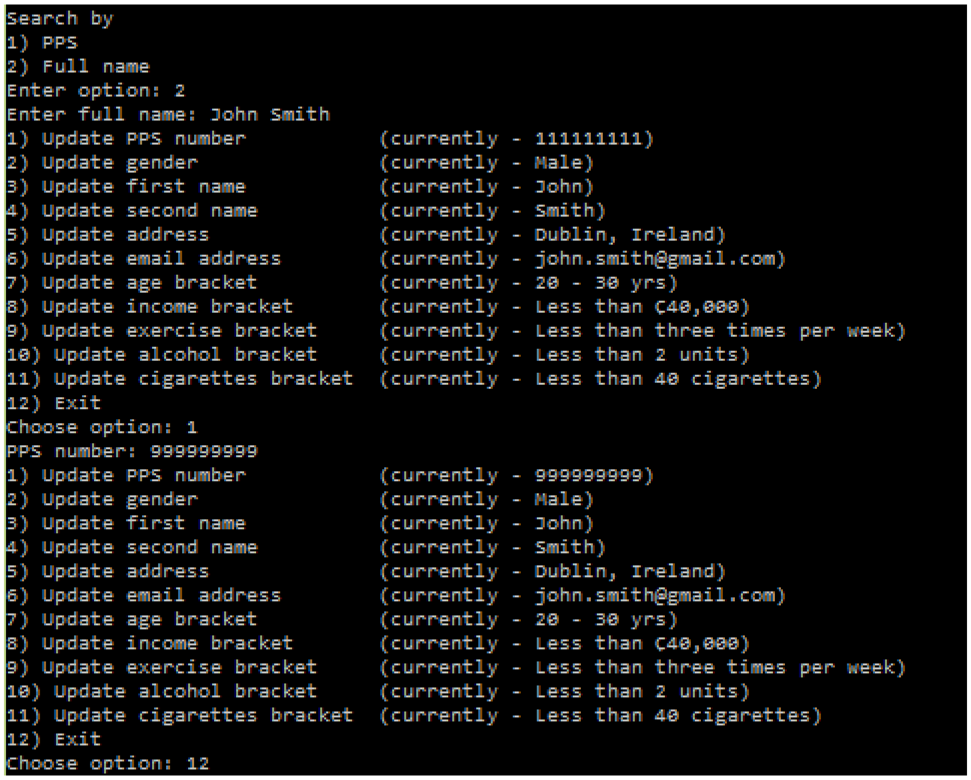
DISPLAY SPECIFIC SURVEY FUNCTION

1. Allow the user to choose which survey to print by entering either the PPS number or full name. This is handled by another function called searchByPPSOrName which returns the index of the survey if it was found, or -1 if it wasn't.
2. If a survey is found, loop through the linked list until the index is reached.
3. Call the displaySurveyDetails and pass the element at that index and stdout to print its details to the console.

UPDATE SURVEY FUNCTION

1. Allow the user to search for a survey to update by either a PPS number or full name.
2. If a survey is found save the element at the index returned from the search in a variable.
3. Similar to the main menu, print a menu with a list of options to choose from, including an option to exit the loop.
4. The other options allow the user to choose which property to choose from.
5. When updating the PPS number, first check if the new PPS number is already in the list.
6. If so print an error message and return to the menu.
7. Otherwise, set the PPS number to the new PPS number, remove the element from the linked list by calling the erase function and add it back by calling the add function. This will reorder the linked list. Note that the erase function does not free the allocated memory.

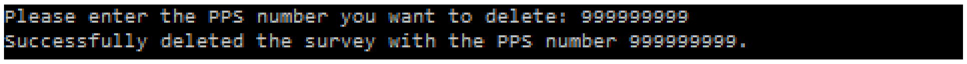
Example of searching for a survey to update via a name and updating the PPS number of that survey.



```
Search by
1) PPS
2) Full name
Enter option: 2
Enter full name: John Smith
1) Update PPS number            (currently - 111111111)
2) Update gender                (currently - Male)
3) Update first name            (currently - John)
4) Update second name           (currently - Smith)
5) Update address               (currently - Dublin, Ireland)
6) Update email address         (currently - john.smith@gmail.com)
7) Update age bracket           (currently - 20 - 30 yrs)
8) Update income bracket        (currently - Less than €40,000)
9) Update exercise bracket      (currently - Less than three times per week)
10) Update alcohol bracket      (currently - Less than 2 units)
11) Update cigarettes bracket   (currently - Less than 40 cigarettes)
12) Exit
Choose option: 1
PPS number: 999999999
1) Update PPS number            (currently - 999999999)
2) Update gender                (currently - Male)
3) Update first name            (currently - John)
4) Update second name           (currently - Smith)
5) Update address               (currently - Dublin, Ireland)
6) Update email address         (currently - john.smith@gmail.com)
7) Update age bracket           (currently - 20 - 30 yrs)
8) Update income bracket        (currently - Less than €40,000)
9) Update exercise bracket      (currently - Less than three times per week)
10) Update alcohol bracket      (currently - Less than 2 units)
11) Update cigarettes bracket   (currently - Less than 40 cigarettes)
12) Exit
Choose option: 12
```

DELETE SURVEY FUNCTION

1. Check if the linked list is empty.
2. If so print an error message.
3. Otherwise, allow the user to search for a survey to delete by entering a PPS number.
4. If no survey matches the search print an error message.
5. Otherwise, remove the element from the linked list at the index returned from the search using the erase function.
6. Free the memory allocated to the removed element.



```
Please enter the PPS number you want to delete: 999999999
Successfully deleted the survey with the PPS number 999999999.
```

GENERATE STATISTICS FUNCTION

1. First declare a 2D array of floats to store all the data needed to calculate the statistics. This array will have 14 rows and 13 columns.

Rows:	Columns:
1. 18 – 20 yrs	1. No. of people who don't smoke
2. 20 – 30 yrs	2. No. of people who smoke less than 20 cigarettes a week
3. 30 – 50 yrs	3. No. of people who smoke less than 40 cigarettes a week
4. 50 – 65 yrs	4. No. of people who smoke more than 40 cigarettes a week

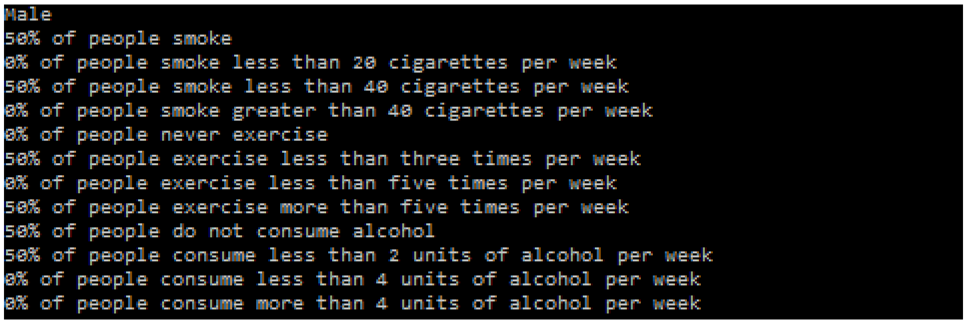
5. 65+ yrs	5. No. of people who never exercise
6. No Income	6. No. of people who exercise less than 3 times a week
7. < €20000	7. No. of people who exercise less than 5 times a week
8. < €40000	8. No. of people who exercise more than 5 times a week
9. < €60000	9. No. of people who don't consume alcohol
10. < €80000	10. No. of people who consume less than 2 units of alcohol per week
11. < €100000	11. No. of people who consume less than 4 units of alcohol per week
12. > €100000	12. No. of people who consume more than 4 units of alcohol per week
13. Male	13. The total number of people who fit the criteria (i.e. the total number of people in the row)
14. Female	

2. Next loop over each element in the linked list and populate the 2D array with the data.
3. The program must output the percentage of people who smoke. However, the values in the first column currently store the number of people who don't smoke. To get the number of people who smoke, loop through the array and take away the number of people who don't smoke from the total number of people in the row, which is the value in the last column.
4. Next use the data from this array to populate a statistics array which is passed to the function when it is called. The statistics array is very similar to the data array except it only has 12 columns. To generate the statistics, divide the value in each of the first twelve columns in the data array by the value in the last column, which is the number of people in the row, and multiply the result by 100 to get the percentage. Store the answer in the corresponding cell of the statistics array.

PRINT STATISTICS FUNCTION

1. The print statistics calls the generateStatistics function.
2. It then loops through the statistics array and prints the statistics for each criteria in a formatted manner to a file passed in as a parameter.

The following screenshot shows the statistics for males based on all three surveys shown in the display all surveys function output. Statistics for all other criteria, meaning age, income and female, are also listed in the same format.



```
Male
50% of people smoke
0% of people smoke less than 20 cigarettes per week
50% of people smoke less than 40 cigarettes per week
0% of people smoke greater than 40 cigarettes per week
0% of people never exercise
50% of people exercise less than three times per week
0% of people exercise less than five times per week
50% of people exercise more than five times per week
50% of people do not consume alcohol
50% of people consume less than 2 units of alcohol per week
0% of people consume less than 4 units of alcohol per week
0% of people consume more than 4 units of alcohol per week
```

GENERATE REPORT FILE FUNCTION

1. Open report.txt file in write mode.
2. If the file cannot be opened print an error message.
3. Otherwise, loop through the linked list of surveys and print each survey to the report file using the displaySurveyDetails function.
4. Next generate the statistics and print the results to the report file using the print statistics function.
5. Finally close the file.

OTHER FUNCTIONS

1. I wrote a getInt and getString method which get a value entered by the user and perform necessary checks on that value. This will prevent code duplication throughout the program.
2. I wrote a function called getNewIndex which will be called when adding an element to the linked list. It loops over the linked list and finds the index to insert a new element at, based on the PPS number. The purpose of this function is to keep the linked list ordered.
3. I wrote an add and erase functions for adding and removing elements from the list since this is common functionality in this program. I initially called the erase function remove but it conflicted with a function in one of the header files included in the program.
4. The verifyEmail function is called when the user is creating or updating an email address. This function performs numerous checks on the email address in order to verify if it is valid. In order to be valid, an email must have a single @ symbol, a full stop before the @ symbol, a domain name after the @ symbol and end in .com. An example of a valid email would be some.name@somedomain.com. If any of the rules are not followed an appropriate error message is printed.