



COMP 1039

Problem Solving and Programming

Programming Assignment 2

Prepared by
Jo Zucco

SAIBT Modifications
Andreas Jordan and Jane Emmett

School of Computer and Information Science
The University of South Australia
June 2017

Contents

INTRODUCTION	1
ASSIGNMENT OVERVIEW	1
PART I SPECIFICATION – WRITING A PYTHON MODULE (LIST MANIPULATION FUNCTIONS	2
PRACTICAL REQUIREMENTS (PART I)	3
STAGES (PART I)	4
Stage 1	4
Stage 2	6
Stage 3	6
PART II SPECIFICATION – MANAGE A SIMPLE SOCIAL NETWORK	7
Interactive Mode	8
PRACTICAL REQUIREMENTS (PART II)	11
STAGES (PART II)	14
Stage 1 – Set up	14
Stage 2 – Importing profile module	14
Stage 3 – Implementing read_file()	15
Stage 4 – Implementing display_summary()	16
Sample Output – Display summary	17
Stage 5 – Implementing write_to_file()	18
Stage 6 – Implementing Interactive menu	18
Stage 7 – Implementing menu options	19
Stage 8 – Validate User input	24
Stage 9 – Comparing your output	24
SUBMISSION DETAILS	25
EXTENSIONS AND LATE SUBMISSIONS	27
ACADEMIC MISCONDUCT	27
DESCRIPTION OF profile.py MODULE	28
Examples – Using methods from Profile class	29
MARKING CRITERIA	31
SAMPLE OUTPUT – PART I	33
SAMPLE OUTPUT – PART II	35
Sample output 1:	35
Sample output 2:	37
Sample output 3:	39
Sample output 4:	41
Sample output 5:	43
Sample output 6:	47
Sample output 7:	50

INTRODUCTION

This document describes the second assignment for Problem Solving and Programming.

The assignment is intended to provide you with the opportunity to put into practice what you have learnt in the course by applying your knowledge and skills to the implementation of a **Python module** (that contains functions that operate on lists) and **a program that will manage a simple social network, by storing and maintaining profile information.**

This assignment is an **individual task** that will require an **individual submission**. **You will be required to present your work to your practical supervisor during your practical session held in Week 12 of the study period.** Important: You must attend the practical session that you have been attending all study period in order to have your assignment marked.

This document is a kind of specification of the required end product that will be generated by implementing the assignment. Like many specifications, it is written in English and hence will contain some imperfectly specified parts. Please make sure you seek clarification if you are not clear on any aspect of this assignment.

ASSIGNMENT OVERVIEW

There are two parts to this assignment:

Part I: Writing a Python Module (list manipulation functions)

You are required to implement a Python module that contains functions that manipulate lists. Please ensure that you read sections titled 'Part I specification' below for further details.

Part II: Manage a simple social network

You are required to write a Python program that will manage a simple social network. The program will store and maintain personal profile information (using a List of Profile objects). Personal profile information will be stored in a text file that will be read in when the program commences. Once the initial profile information has been read in from the file, the program should allow the user to interactively query and manipulate the profile information. Please ensure that you read sections titled 'Part II specification' below for further details.

Please ensure that you read sections titled 'Part I Specification' and 'Part II Specification' below for further details.

PART I SPECIFICATION – WRITING A PYTHON MODULE (LIST MANIPULATION FUNCTIONS)

You are required to write a `list_function.py` module (containing only the functions listed below). This file is provided for you (on the course website) however, you will need to modify this file by writing code that implements the functions listed below. **Please read the slides on modules available on the course website if you would like more information on modules.**

You are required to implement a Python module containing the following functions:

1. Write a function called `length(my_list)` that takes a list as a parameter and returns the length of the list. You must use a loop in your solution. You **must not** use built-in functions, list methods or string methods in your solution.
2. Write a function called `to_string(my_list, sep=', ')` that takes a list and a separator value as parameters and returns the string representation of the list (separated by the separator value) in the following form:

```
item1, item2, item3, item4
```

The separator value **must** be a default argument. i.e. `sep=', '`

You must use a loop in your solution. You **must not** use built-in functions (other than the `range()` and `str()` functions), slice expressions, list methods or string methods in your solution. You may use the concatenation (+) operator to build the string. You **must** return a string from this function.

3. Write a function called `count(my_list, value)` that takes a list and a value as parameters. The function searches for the value in the list and returns how many times the value appears in the list. You may assume that the elements of the list can be compared using the comparison operators `==`, `!=`, etc. You must use a loop in your solution. You **must not** use built-in functions (other than the `range()` function), list methods or string methods in your solution.
4. Write a function called `find(my_list, value)` that takes a list, and a value as parameters. The function searches for the value in the list and returns the index at which the first occurrence of value is found in the list. The function returns -1 if the value is not found in the list.
5. Write a function called `insert_value(my_list, value, insert_position)` that takes a list, a value and an `insert_position` as parameters. The function returns a copy of the list with the value inserted into the list (`my_list`) at the index specified by `insert_position`. Check for the `insert_position` value exceeding the list (`my_list`) bounds. If the `insert_position` is greater than the length of the list, insert the value at the end of the list. If the `insert_position` is less than or equal to zero, insert the value at the start of the list. You **must** use a loop(s) in your solution. You may make use of the list name `.append(item)` method in order to build the new list. You **must not** use built-in functions (other than the `range()` function), slice expressions, list methods (other than the `append()` method) or string methods in your solution.
6. Write a function called `remove_value(my_list, remove_position)` that takes a list and a `remove_position` as parameters. The function returns a copy of the list with the item at the index specified by `remove_position`, removed from the list. Check for the `remove_position` value exceeding the list (`my_list`) bounds. If the `remove_position` is greater than the length of the list, remove the item at the end of the list. If the `remove_position` is less than or equal to zero, remove the item stored at the start of the list. You **must** use a loop in your solution. You may make use of the list name `.append(item)` method in order to build the new list. You **must not** use built-in functions (other than the `range()` function), slice expressions, list methods (other than the `append()` method) or string methods in your solution.

You must test your functions to ensure that they are working correctly. **So you do not have to write your own test file, one has been provided for you.** The `assign2_partI_test_file.py` file is a test file that contains code that calls the functions contained in the `list_function.py` module. **Please do not modify the test file.**

PRACTICAL REQUIREMENTS (PART I)

It is recommended that you develop this part of the assignment in the suggested stages.

It is expected that your solution will include the use of:

- The supplied `list_function.py` module (containing the functions listed below). This is provided for you – you will need to modify this file.
- Functions (`length`, `to_string`, `count`, `find`, `insert_value`, and `remove_value`) implemented adhering to the assignment specifications.
- The supplied `assign2_partI_test_file.py` file. This is provided for you – **please DO NOT modify this file**.
- Well constructed while loops. (Marks will be lost if you use `break` statements in order to exit from loops).
- Well constructed for loops. (Marks will be lost if you use `break` statements in order to exit from loops).
- Appropriate `if/elif/else` statements.
- Output that **strictly** adheres to the assignment specifications. If you are not sure about these details, you should check with the 'Sample Output – Part I' provided at the end of this document.
- Good programming practice:
 - Consistent commenting and code layout. You are to provide comments to describe: your details, program description, all variable definitions, all functions, and every significant section of code.
 - Meaningful variable names.
- Your solutions **MAY** make use of the following:
 - Built-in functions `range()` and `str()`.
 - List method `append()` to create/build new lists. i.e. `list_name.append(item)`.
 - Concatenation (+) operator to create/build new strings.
 - Comparison operators (`==`, `!=`, `<`, `>`, etc).
 - Access the individual elements in a list with an index (one element only). i.e. `list_name[index]`.
 - Use of any of the functions you have written as part of the assignment. i.e. `length()` function.

Your solutions **MUST NOT** use:

- Built-in functions (other than `range()` and `str()` functions).
- Slice expressions to select a range of elements from a list. i.e., `list_name[start:end]`.
- List methods (other than the `append()` method. i.e. `list_name.append(item)`).
- String methods.
- Do not use `break`, or `continue` statements in your solution – doing so will result in a significant mark deduction. Do not use the `quit()` or `exit()` functions as a way to break out of loops.

It is recommended that you use Python 3.6.0 (or most current version) in order to complete your assignments. Your programs **MUST** run using Python 3.6.0 (or most current version).

STAGES (PART I)

It is recommended that you develop this part of the assignment in the suggested stages. Many problems in later stages are due to errors in early stages. **Make sure you have finished and thoroughly tested each stage before continuing.**

The following stages of development are recommended:

Stage 1

You will need both the `list_function.py` and `assign2_partI_test_file.py` files for this assignment. These have been provided for you. Please download both of these files from the course website and ensure that they are in the same directory as each other.

Test to ensure that this is working correctly by opening and running the `assign2_partI_test_file.py` file. If this is working correctly, you should now see the following output in the Python shell when you run your program:

```
Start Testing!

length Test

In function length()
List length: None
In function length()
List length: None

to_string Test
In function to_string()
List is: None
In function to_string()
List is: None In function to_string()
List is: None

count Test
In function count()
None
In function count()
None
In function count()
None

find Test
In function find()
None
In function find() None
insert_value Test
In function insert_value()
None
In function insert_value()
None
In function insert_value()
None
In function insert_value()
None

remove_value Test
In function remove_value()
None
In function remove_value()
```

None
In function remove_value()
None

length Test
In function length()
List length: None

to_string Test
In function to_string()
List is: None
In function to_string()
List is: None

count Test
In function count()
None

find Test
In function find()
None

insert_value Test
In function insert_value()
None

remove_value Test
In function remove_value()
None

End Testing!

Stage 2

Implement one function at a time. The following implementation order is a recommendation only:

- `length()`
- `to_string()`
- `count()`
- `find()`
- `remove_value()`
- `insert_value()`

Place the code that implements each function in the appropriate place in the `list_function.py` file.

For example, if you were implementing the `length()` function, you would place the code that calculates and returns the length of the list under the comment 'Place your code here' (within the `length` function definition) seen below.

```
# Function length() – place your own comments here... :)
def length(my_list):

    # This line will eventually be removed – used for
    # development purposes only.
    print('In function length()')
    # Place your code here
```

Test your function by running the `assign2_partI_test_file.py` test file to ensure each function is working correctly before starting on the next function.

Compare your output with the sample output provided (at the end of this document) to ensure that your function is working as it should.

Stage 3

Finally, check the sample output (see section titled 'Sample Output – Part I' towards the end of this document) and if necessary, modify your functions so that:

- The output produced by your program **EXACTLY** adheres to the sample output provided.
- Your program behaves as described in these specs and the sample output provided.

PART II SPECIFICATION – MANAGE A SIMPLE SOCIAL NETWORK

Write a **menu driven program** called `yourSaibtId.social.py` that will allow the user to enter commands and process these commands until the quit command is entered. The program will store and maintain personal profile information (using a List of Profile objects). Personal profile information will be stored in a text file that will be read in when the program commences. Once the initial profile data has been read in from the file, the program should allow the user to interactively query and manipulate the profile information.

Input

When your program begins, it will read in personal profile information from a file called `profiles.txt`. This is a text file that stores profile information for the simple social network. An example input file called `profiles.txt` is provided on the course website (under the Assessment tab). You may assume that all data is in the correct format. Details for each person is listed as follows:

- 1st line – Given name, family name, email address, and gender are stored on one line (separated by a space)
- 2nd line – Person's current status
- 3rd line – A single number that represents how many friends the person has. For example, if the number is 0, the following line will be the start of a new profile. If the number is 1, then the following line is a single email address of the friend before starting a new profile on the line after that.

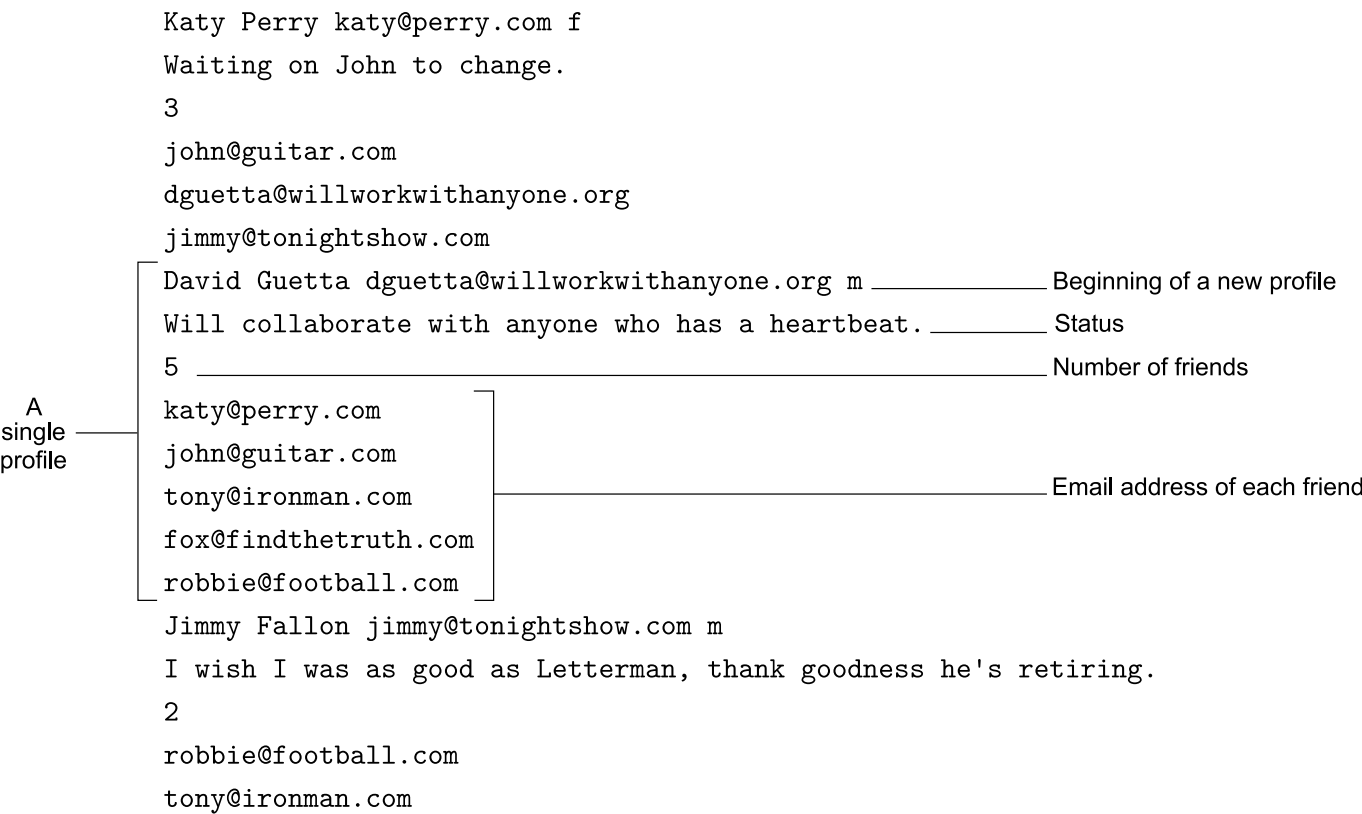


Figure 1: Extract from supplied file – `profiles.txt`

After the program has stored the data (using a List of Profile objects), it will enter interactive mode as described in the following section.

Interactive Mode

Your program should enter an interactive mode after the profile information has been read from the file. The program will allow the user to enter commands and process these commands until the quit command is entered. The following commands should be allowed:

1. Summary:

Outputs the contents of the profile list as seen below in the section titled *Screen Format*.

2. Add:

Prompts for and reads a person's email address. If the email address does not already exist (i.e. a match is not found on email address) in the profile list, prompts for and reads the rest of the person's details (given name, family name, gender and status) and adds the information to the profile list (note that the number of friends will be set to zero – no friends are read in at this point). A message is displayed to the screen indicating that the profile has been successfully added. The profile must be added after the last profile entry stored in the list. If the profile is already stored in the profile list, an error message is displayed.

```
Please enter choice [summary|add|remove|search|update|quit]: add
```

```
Please enter email address: homer@doh.com
Please enter given name: Homer
Please enter family name: Simpson
Please enter gender: m
Please enter current status: mmmmmmm.... donuts.... :)
```

```
Successfully added homer@doh.com to profiles.
```

```
Please enter choice [summary|add|remove|search|update|quit]: add
```

```
Please enter email address: john@guitar.com
john@guitar.com already exists in profiles.
```

3. Remove:

Prompts for and reads the person's email address. If the email address (profile) is found, it is removed from the list of profiles and a message is displayed to the screen indicating that this has been done. If the profile is not found in the profiles list, an error message is displayed. Note: The deleted person's email address must also be removed from ALL profiles that are friends with the deleted profile. That is, remove the email address from all friends' lists.

Sample Output 1

```
Please enter choice [summary|add|remove|search|update|quit]: remove
Please enter email address: robbie@football.com
```

```
Successfully removed robbie@football.com from profiles.
```

Sample Output 2

```
Please enter choice [summary|add|remove|search|update|quit]: remove
Please enter email address: bruce.wayne@batcave.com
```

```
bruce.wayne@batcave.com is not found in profiles.
```

4. Search:

Prompts for and reads the person's email address and searches for the person in the profile list. If the person is found in the profile list, the person's details are displayed to the screen as seen below in the section titled *Screen Format*. If the person is not found in the profile list, an error message stating the person has not been found is displayed.

```
Please enter choice [summary|add|remove|search|update|quit]: search
Please enter email address: john@guitar.com
```

```
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
  katy@perry.com
  dguetta@willworkwithanyone.org
```

5. Update:

Prompts for and reads the person's email address. If the email (profile) is not found in the profiles list, an error message is displayed to the screen.

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: brad@pitt.com
```

```
brad@pitt.com was not found in profiles.
```

If the profile with matching email address is found, the following prompt is displayed:

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: robbie@football.com
Update Robbie Gray [status|add_friend|remove_friend]:
```

- **status** – Prompt for and read the new status; update the status accordingly.
- **add_friend** – Prompt for and read the email address of the friend to add.
 - If the friend's email address is not found in the list of profiles, display an error message to the screen.
 - If the email address is found in the friends list (i.e. they are already friends), display an error message to the screen (duplicate entries are not allowed).
 - Otherwise, add the email address to the friends list and display a message to the screen indicating that this has been done.
- **remove_friend** – Prompt for and read the email address of the friend to remove.
 - If the friend's email address is not found in the list of friends, display an error message to the screen.
 - If the email address is found in the friends list, it is removed from the array of friends and a message is displayed to the screen indicating that this has been done.

6. Quit:

Causes the program to quit and output the contents of the list of profiles to a file called `new_profiles.txt`. The format of this file should exactly match that of the input file.

Note:

The program should display an appropriate message if a profile is not found matching a search criteria. Appropriate messages should also be displayed to indicate whether a command has been successfully completed.

Please refer to the sample output (at the end of this handout) to ensure that your program is behaving correctly and that you have the correct output messages.

Screen Format

The `summary` command (`display_summary()` function) should display the profile information in the following format:

```
=====
Profile Summary
=====
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
Tony Stark
-----
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
Robbie Gray
Fox Mulder
-----
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
Katy Perry
David Guetta
-----
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
John Mayer
David Guetta
Jimmy Fallon
-----
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
Katy Perry
John Mayer
Tony Stark
Fox Mulder
Robbie Gray
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
Robbie Gray
Tony Stark
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
Jimmy Fallon
Fox Mulder
John Mayer
Tony Stark
=====
```

PRACTICAL REQUIREMENTS (PART II)

It is recommended that you develop this part of the assignment in the suggested stages. Each stage is worth a portion of the marks.

It is expected that your solution will include the use of:

- Your solution in a file called `yourSaibtId.converter.py`.
- The supplied `profile.py` module (that defines the `Profile` class). This is provided for you – **do NOT modify this file**.
- Appropriate and well constructed `while` and/or `for` loops. (Marks will be lost if you use `break` statements in order to exit from loops).
- Appropriate `if`, `if–else`, `if–elif–else` statements (as necessary).
- The following functions:

- `read_file(filename, profile_list)`

- Takes the name of the input file and list of profile objects(`profile_list`) as parameters
- Reads the contents of the file into the `profile_list(list)`
- Returns the list of profile objects
- You must use a loop in your solution. You **may** use String and/or List methods in this function only. You may find the String methods `split()` and `strip()` useful here.

- `write_to_file(filename, profile_list)`

- Accepts the name of the output file (`filename`) and list of profile objects (`profile_list`) as parameters
- Output the contents of the profile list (list of profile objects) to a file **in the same format as the input file**
- The file will need to be opened for writing in this function (and of course closed once all writing has been done).
- The function returns nothing
- You must use a loop in your solution.

- `display_summary(profile_list)`

- Accepts a list of profile objects as a parameter
- Displays the contents of the list to the screen in the format specified (see *Screen Format* on page 10)
- The function returns nothing
- You must use a loop in your solution

- `find_profile(profile_list, email)`

- Accepts a person's email as input along with the list of profile objects (`profile_list`) as parameters
- Returns the position (`index`) of the person found in the `profile_list`. If the person is not found, the function returns `–1`.
- You **must** use a loop in your solution. You **must not** use list methods in your solution.

- `add_profile(profile_list)`
 - Accepts a list of profile objects as a parameter
 - Prompts for and reads the person's (to be added) email address
 - ▷ If the person already exists in the list of profiles, display an error message to the screen
 - ▷ If the person does not exist in the profiles list, prompt for and read the person's given name, family name, gender and status
 - ▷ Create a new profile object with the information read in and add the profile object to the end of the list of profiles and displays a message to the screen indicating that a new person has been added to the profiles list
 - ▷ Returns the list of profiles
 - ▷ You may use the `list_name.append(item)` method to add the new profile object to the list of profiles. You must call function `find_profile()` from this function
- `remove_profile(profile_list)`
 - Accepts a list of profile objects (i.e., `profile_list`) as a parameter
 - Prompts for and reads the person's email address (to be removed)
 - ▷ If the person is not found in the list of profiles, display an error message to the screen
 - ▷ If the person does exist in the profiles list, this function removes the person's profile object and displays a message to the screen indicating that the person has been removed from the profiles list
 - ▷ Note: The deleted person's email address must also be removed from ALL profiles that are friends with the deleted profile. That is, remove the email address from all friends' lists (you may make use of the `profile.remove_friend(email)` method to do this)
 - ▷ Returns a list of profiles
 - ▷ You may use the `list_name.append(item)` method in this function. You **must** call function `find_profile()` from this function

Your solutions **MAY** make use of the following:

- Built-in functions `int()`, `input()`, `print()`, `range()`, `open()`, `close()`, `len()` and `str()`.
- Concatenation (+) operator to create/build new strings.
- Access the individual elements in a string with an index (one element only). i.e. `string_name[index]`.
- Access the individual elements in a list with an index (one element only). i.e. `list_name[index]`.
- Profile objects and methods (as appropriate).
- The `list_function.py` module (that you wrote in part I of this assignment). You may like to make use of some of the functions defined in the `list_function.py` module for this part of the assignment (as appropriate). Not all will be suitable or appropriate.

Your solutions **MUST NOT** use:

- Built-in functions (other than the `int()`, `input()`, `print()`, `range()`, `open()`, `close()`, `len()` and `str()` functions).
- Slice expressions to select a range of elements from a string or list. i.e. `name[start:end]`.
- String or list methods (i.e., other than those mentioned in the 'MAY make use' of section above).
- Global variables as described in week 8 lecture.
- The use of `break`, `return` or `continue` statements (or any other technique to break out of loops) in your solution – doing so will result in a significant mark deduction.

PLEASE NOTE: You are reminded that you should ensure that all input and output conform to the specifications. If you are not sure about these details you should check with the sample output provided at the end of this document or post a message to the discussion forum in order to seek clarification.

Please ensure that you use Python 3.6.0 or a later version (i.e. the latest version) in order to complete your assignments. Your programs **MUST** run using Python 3.6.0 (or latest version).

STAGES (PART II)

It is recommended that you develop this part of the assignment in the suggested stages. Many problems in later stages are due to errors in early stages. **Make sure you have finished and thoroughly tested each stage before continuing.**

The following stages of development are recommended:

Stage 1 – Set up

To begin, download the following provided files (available on the course website):

- `profile.txt`
- `profile.py` – See page xxx for a description of available functions in this file
- `social.py`
 - Rename `social.py` to `yourSaibtId_social.py`
 - Define an empty list to store the profile information. For example:
`profile_list = []`

Stage 2 – Importing profile module

Create a profile object from class `Profile` (provided for you in the `profile.py` module).

- Open `social.py` in IDLE and *uncomment* the following line at the top of your program:

```
import profile
```

- Create and display the profile object below inside your `read_file()` function

```
new_profile = profile.Profile("Bruce", "Wayne", "batman@batcave.com", "m", "fighting crime")
new_profile.set_friends_list(["tony@ironman.com", "superman@upupandaway.com"])
print(new_profile)
```

- Make sure the program runs correctly. You should see the following output:

```
Bruce Wayne batman@batcave.com m
fighting crime
2
tony@ironman.com
superman@upupandaway.com
```

- Once you are sure that your program is working correctly, you can delete the above two statements – we just wanted to make sure it was working correctly!

Once you have that working, back up your program. *Note: When developing software, you should always have fixed points in your development where you know your software is bug free and runs correctly.*

Stage 3 – Implementing `read_file()`

Write the code for function `read_file()` following the algorithm provided below:

```
1  def read_file(filename, profile_list)
2      Open file
3      Read line
4      WHILE line not end of file DO
5          line→Strip newline character from end of line
6          Create new profile
7          Split line into list→given name, family name, email, gender
8          Add details to new profile
9          Read line → get Status
10         line→Strip newline character from end of line
11         Add status to new profile
12         Read line → get number of friends
13         line→Strip newline character from end of line
14         Add number to new profile (i.e. set_number_of_friends())
15         Create an empty list to store friends
16         FOR index in 0..No. of friends DO
17             Read line → friend
18             line→Strip newline character from end of line
19             APPEND friend to friend list
20         Add friends list to new profile
21         Append profile to profile_list
22         Read line
23         #End of main loop
```

Add code to call the function to ensure it is working correctly.

Examples

Split()

The following code splits the string `user_details` into a list, basing the **split** on the space character:

```
line = 'Bryan Adams bryan@singer.com m'
user_details = line.split(' ')
for user_detail in user_details:
    print(user_detail)
```

Produces the following output:

```
Bryan
Adams
bryan@singer.com
m
```

`strip()`

The following code removes the `'\n'` from the end of the string:

```
line = 'Bryan Adams bryan@singer.com m\n'
line.strip('\n')
print(line)
```

Produces the following output:

```
Bryan Adams bryan@singer.com m
```

Stage 4 – Implementing `display_summary()`

Write the code for function `display_summary()`. Add code to call the function to ensure it is working correctly.

- At this stage, you will have each contact from `profiles.txt` read in and inserted (as individual profiles) into `profile_list`. Each profile represents an **object**, in other words, an *instance* of the profile class.
- These individual instances are stored in a list (object) called `profile_list` which you should already have implemented and used to add profiles in [Stage 3](#).
- You'll need a loop and use the methods in the profile class to obtain the different data attributes for each instance, e.g. `given_name`, `status`, `email`, etc. For more examples of using methods in the profile class, see page 29.

```
1 for person in profile_list:
2     print(person.get_given_name() + " " + person.get_family_name())
```

Sample Output – Display summary

Profile Summary

Fox Mulder (m | fox@findthetruth.com)

- The truth is out there!

- Friends (1):

Tony Stark

Tony Stark (m | tony@ironman.com)

- Saving the world is hard work - no time for friends.

- No friends yet...

Phil Dunphy (m | phil@dunphy.com)

- wtf? = why the face?

- Friends (2):

Robbie Gray

Fox Mulder

John Mayer (m | john@guitar.com)

- Waiting on the world to change!

- Friends (2):

Katy Perry

David Guetta

Katy Perry (f | katy@perry.com)

- Waiting on John to change.

- Friends (3):

John Mayer

David Guetta

Jimmy Fallon

David Guetta (m | dguetta@willworkwithanyone.org)

- Will collaborate with anyone who has a heartbeat.

- Friends (5):

Katy Perry

John Mayer

Tony Stark

Fox Mulder

Robbie Gray

Jimmy Fallon (m | jimmy@tonightshow.com)

- I wish I was as good as Letterman, thank goodness he's retiring.

- Friends (2):

Robbie Gray

Tony Stark

Robbie Gray (m | robbie@football.com)

- Training hard... can we win? Yes we Ken!

- Friends (4):

Jimmy Fallon

Fox Mulder

John Mayer

Tony Stark

Stage 5 – Implementing `write_to_file()`

Now that you know the information is being correctly stored in your profile lists, write the code for function `write_to_file()`. Add code to call this function to ensure it is working correctly.

Note – Your output should look precisely like the contents of `profiles.txt`

Stage 6 – Implementing Interactive menu

Implement the interactive mode, i.e. to prompt for and read menu commands. Set up a loop to obtain and process commands. Test to ensure that this is working correctly before moving onto the next stage. You do not need to call any functions at this point, you may simply display an appropriate message to the screen, for example:

Sample output:

```
Please enter choice [summary|add|remove|search|update|quit]: roger
Not a valid command - please try again.

Please enter choice [summary|add|remove|search|update|quit]: summary
In summary command

Please enter choice [summary|add|remove|search|update|quit]: add
In add command

Please enter choice [summary|add|remove|search|update|quit]: remove
In remove command

Please enter choice [summary|add|remove|search|update|quit]: search
In search command

Please enter choice [summary|add|remove|search|update|quit]: update
In update command

Please enter choice [summary|add|remove|search|update|quit]: quit
```

Menu input should be validated with an appropriate message being displayed if incorrect input is entered by the user.

Stage 7 – Implementing menu options

Implement one command at a time. Test to ensure the command is working correctly before starting the next command. Start with the `quit` and `summary` commands as they do not need you to add anything further to the file other than ensuring that the function calls are in the correct place.

You should be able to see that for most commands there is a corresponding function(s).

For the remaining commands, the following implementation order is suggested (note: this is a guide only):

1. `summary` command (`display_summary()` function).

2. `search` command (`find_profile()` function).

Hint: Make use of the following functions from `profile.py`

- `profile.get_email()`

Sample Output 1 – Find profile

```
Please enter choice [summary|add|remove|search|update|quit]: search
Please enter email address: elvis@allshookup.org
elvis@allshookup.org is not found in profiles.
Please enter choice [summary|add|remove|search|update|quit]:
```

Sample Output 2 – Find profile

```
Please enter choice [summary|add|remove|search|update|quit]: search

Please enter email address: dguetta@willworkwithanyone.org

David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
  katy@perry.com
  john@guitar.com
  tony@ironman.com
  fox@findthetruth.com
  robbie@football.com

Please enter choice [summary|add|remove|search|update|quit]:
```

add command (add_profile() function).

Hint: Make use of the following functions from profile.py

- profile.set_given_name(given_name)
- profile.set_family_name(family_name)
- profile.set_gender(gender)
- profile.set_status(status)

Sample Output 1 – Add profile

```
Please enter choice [summary|add|remove|search|update|quit]: add
Please enter email address: john@guitar.com
john@guitar.com already exists in profiles.
```

Sample Output 2 – Add profile

```
Please enter choice [summary|add|remove|search|update|quit]: add

Please enter email address: homer@doh.com
Please enter given name: Homer
Please enter family name: Simpson
Please enter gender: m
Please enter current status: mmmmmmm.... donuts.... :)

Successfully added homer@doh.com to profiles.
```

remove command (remove_profile() function).

Hint: Make use of the following functions from profile.py

- profile.get_email()
- profile.remove_friend(email)

Sample Output 1 – Remove profile

```
Please enter choice [summary|add|remove|search|update|quit]: remove

Please enter email address: bruce.wayne@batcave.com
bruce.wayne@batcave.com is not found in profiles.
```

Sample Output 2 – Remove profile

```
Please enter choice [summary|add|remove|search|update|quit]: remove

Please enter email address: robbie@football.com
Successfully removed robbie@football.com from profiles.
```

update command.

The update command has three(3) sub-options:

- status - Update status of a person's profile
- add_friend – Add a friend to the person's profile
- remove_friend – Remove a friend from the person's profile

Hint: Make use of following functions from `profile.py`

- `profile.set_friends_list(aList)`
- `profile.get_friends_list(aList)`
- `profile.get_friends_list()`
- `profile.set_status(new_status)`
- `profile.remove_friend(previous_friend)`
- `profile.get_email()`
- `profile.get_given_name()`
- `profile.get_family_name()`

The pseudocode for the function `update_profile` has been provided on page 22

```

1  def UPDATE PROFILE(profile_list)
2      FRIEND_NOT_FOUND → True
3      email → PROMPT user enter EMAIL ADDRESS
4      is_found → FALSE
5      for profile in profile_list DO
6          if profile.get_email() EQUALS email THEN
7              is_found → True
8      if is_found THEN
9          For EACH profile in profile_list DO
10             profile_email → profile.get_email()
11             IF email EQUALS profile_email THEN
12                 action → input('Update ' + profile.get_given_name()
13                     + ' ' + profile.get_family_name()
14                     + ' ' + '[status|add_friend|remove_friend]: ')
15
16             IF action EQUALS 'status' THEN
17                 new_status → PROMPT user to 'Please new status: '
18                 profile.set_status(new_status)
19
20             ELSE IF action EQUALS 'add_friend' THEN
21                 new_friend → PROMPT user enter EMAIL ADDRESS
22                 friends → profile.get_friends_list()
23                 FOR friend in friends DO
24                     IF friend EQUALS new_friend THEN
25                         DISPLAY 'You are already friends this person (duplicate entries not permitted)'
26                         FRIEND_NOT_FOUND → False
27                     IF FRIEND_NOT_FOUND THEN
28                         APPEND new_friend TO friends LIST
29                         DISPLAY new_friend + 'has been added to your friends list.'
30                         profile.set_friends_list(friends)
31
32             ELSE IF action EQUALS 'remove_friend'
33                 previous_friend → PROMPT user to 'Enter email address: '
34                 friends → profile.get_friends_list()
35                 FOR friend in friends DO
36                     IF friend EQUALS previous_friend THEN
37                         profile.remove_friend(previous_friend)
38                         FRIEND_NOT_FOUND → FALSE
39                     IF FRIEND_NOT_FOUND THEN
40                         DISPLAY 'You are not friends with this person Remove request ignored.!'
41                     ELSE
42                         DISPLAY 'Friend' + previous_friend + 'removed'
43                 ELSE
44                     PROMPT 'Not a valid command – please try again.'
45
46 ELSE
47     DISPLAY email + 'is not found in profiles.'

```

Table 1: Pseudocode for update_profile

Sample Output 1 – Add friend

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: robbie@football.com
Update Robbie Gray [status|add_friend|remove_friend]: add_friend
Please enter email address of friend to add: katy@perry.com

Added Katy updated profile is:
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (5):
  jimmy@tonightshow.com
  fox@findthetruth.com
  john@guitar.com
  tony@ironman.com
  katy@perry.com
```

Sample Output 2 – Add friend — email not found

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: tony@ironman.com
Update Tony Stark [status|add_friend|remove_friend]: add_friend
Please enter email address of friend to add: homer@doh.com
homer@doh.com is not found in profiles.
```

Sample Output 3 – Add friend — email already exists

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: robbie@football.com
Update Robbie Gray [status|add_friend|remove_friend]: add_friend
Please enter email address of friend to add: john@guitar.com
John is already a friend.
```

Sample Output 4 – Add friend — Invalid command

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: robbie@football.com
Update Robbie Gray [status|add_friend|remove_friend]: add
Not a valid command - returning to main menu.
```

Stage 8 – Validate User input

Ensure that you have validated all user input with an appropriate message being displayed for incorrect input entered by the user. Add code to validate all user input. Hint: use a while loop to validate input.

Stage 9 – Comparing your output

Finally, check the sample output (see section titled ‘Sample Output – Part II towards the end of this document) and if necessary, modify your code so that:

- The output produced by your program **EXACTLY** adheres to the sample output provided.
- Your program behaves as described in these specs and the sample output provided.

SUBMISSION DETAILS

You are required to demonstrate your assignment to your practical supervisor during your week 12 class for marking. The supervisor will mark your work using the marking criteria included in this document. You MUST attend the practical session that you have been attending all study period in order to have your assignment marked.

You are also required to submit an electronic copy of your program via Moodle. Assignments submitted to Moodle, but not demonstrated during your allocated practical session, will NOT be marked. Likewise, assignments that have been demonstrated during the practical session, but have not been submitted via Moodle, will NOT be marked. Assignments are submitted to Moodle in order to check for plagiarism.

All students must follow the submission instructions below:

Ensure that your files are named correctly (as per instructions outlined in this document).

Ensure that the following files are included in your submission:

- `list_function.py`
- `assign2_partI_test_file.py` – this will not be modified, but you should submit it anyway.
- `profile.py` – this will not be modified, but you should submit it anyway.
- `yourSAIBTIId.social.py`

- `profiles.txt`

For example:

- `list_function.py`
- `assign2_partI_test_file.py`
- `profile.py`
- `bonjy007.social.py`
- `profiles.txt`

All files that you submit must include the following comments.

```
#  
# File: fileName.py  
# Author: your name  
# Saibt Id: your saibt id  
# Description: Assignment 2 – place assignment description here...  
# This is my own work as defined by the University's  
# Academic Misconduct policy.  
#
```

Assignments that do not contain these details may not be marked.

You must submit your program **at the start of class before** you demonstrate the work to your marker. You will also be required to demonstrate that you have correctly submitted your work to Moodle. Work that has not been correctly submitted to Moodle will not be marked.

It is expected that students will make copies of all assignments and be able to provide these if required.

EXTENSIONS AND LATE SUBMISSIONS

There will be no extensions/late submissions for this course without one of the following exceptions:

1. A medical certificate is provided that has the timing and duration of the illness and an opinion on how much the student's ability to perform has been compromised by the illness. Please note if this information is not provided the medical certificate **WILL NOT BE ACCEPTED**. Late assessment items will not be accepted unless a medical certificate is presented to the Course Coordinator. The certificate must be produced as soon as possible and must cover the dates during which the assessment was to be attempted. In the case where you have a valid medical certificate, the due date will be extended by the number of days stated on the certificate up to five working days.
2. A SAIBT counsellor contacts the Course Coordinator on your behalf requesting an extension. Normally you would use this if you have events outside your control adversely affecting your course work.
3. Unexpected work commitments. In this case, you will need to attach a letter from your work supervisor with your application stating the impact on your ability to complete your assessment.
4. Military obligations with proof.

Applications for extensions must be lodged via learnonline before the due date of the assignment.

Note: Equipment failure, loss of data, 'Heavy work commitments' or late starting of the course are not sufficient grounds for an extension.

ACADEMIC MISCONDUCT

Students are reminded that they should be aware of the academic misconduct guidelines available from the SAIBT website (see <https://www.saibt.sa.edu.au/policies>).

Deliberate academic misconduct such as plagiarism is subject to penalties.

DESCRIPTION OF `profile.py` MODULE

The `profile.py` module provides a profile class definition called `Profile`. This class definition is to be used to create profile objects and is designed to be used for assignment 2 (part II) work. Profile objects will be used to store a person's personal profile information.

To make use of class `Profile` defined within the profile module, you will need to import the profile module. This should have been done in Stage 1 (i.e. uncommenting the line `#import profile` in `social.py`).

Profile Object Methods

Method	Description
<code>set_given_name(name)</code>	Sets the person's given name.
<code>get_given_name()</code>	Returns the person's given name.
<code>set_family_name(name)</code>	Sets the person's family name.
<code>get_family_name()</code>	Returns the person's family name.
<code>set_email(email)</code>	Sets the person's email address.
<code>get_email()</code>	Returns the person's email address.
<code>set_gender(gender)</code>	Sets the person's gender.
<code>get_gender()</code>	Returns the person's gender.
<code>set_status(status)</code>	Sets the person's status.
<code>set_status()</code>	Returns the person's status.
<code>set_number_friends(no_friends)</code>	Sets the number of friends the person is friends with.
<code>get_number_friends()</code>	Returns the number of friends the person is friends with.
<code>set_friends_list(friends_list)</code>	Sets the friends list of email addresses.
<code>get_friends_list()</code>	Returns the friends list of email addresses.
<code>__str__()</code>	Returns the string representation of the <code>Profile</code> object.
<code>add_friend(email)</code>	Adds an email address to the person's list of friends (<code>friends_list</code>), only if the email doesn't already exist. No duplicate entries are allowed. Returns <code>True</code> if successful and <code>False</code> otherwise.
<code>remove_friend(email)</code>	Removes an email address from the person's list of friends (<code>friends_list</code>). Returns <code>True</code> if successful and <code>False</code> otherwise.
<code>is_friend(email)</code>	Determines whether the email passed in as a parameter exists in the <code>friends_list</code> , i.e. they are friends. Returns <code>True</code> if the email is found in the list of friends (<code>friends_list</code>) and <code>False</code> otherwise.

Table 2: Methods in `Profile` class

Examples – Using methods from Profile class

You can construct a `Profile` object and call it's methods as seen in the following example:

```
import profile

# Create Profile object with given name 'Bruce', family name 'wayne',
# email address 'batman@batcave.com',
# gender 'm' and status as 'fighting crime'.
new_profile1 = profile.Profile("Bruce", "Wayne", "batman@batcave.com",
                              "m", "fighting crime")

# Display the Profile object to the screen as specified by the __str__ method.
print(new_profile1)

# Create another Profile object
new_profile2 = profile.Profile("Tony", "Stark", "tony@ironman.com", "m",
                              "talk to the hand")

# Add Tony Stark as friend of Bruce Wayne
new_profile1.add_friend("tony@ironman.com")

# Create yet another Profile object
new_profile3 = profile.Profile("Clark", "Kent", "superman@upupandaway.com", "m",
                              "flying high")

# Add superman as friend of Bruce Wayne
new_profile1.add_friend("superman@upupandaway.com")

# Add superman as friend of Bruce Wayne
new_profile3.add_friend("tony@ironman.com")

# Display the Profile object to the screen as specified by the __str__ method.
print(new_profile1)

# Display the Profile object to the screen as specified by the __str__ method.
print(new_profile3)

# Update Clark Kent's status
new_profile3.set_status("kryptonite kaos")

# Display the Profile object to the screen as specified by the __str__ method.
print(new_profile3)

# Create list of superheroes (Profile objects)
superhero_list = []
```

```

# Append profile objects to list
superhero_list.append(new_profile1)
superhero_list.append(new_profile2)
superhero_list.append(new_profile3)
print("\n\nDisplaying superhero_list:")

# Iterate over superhero list and display secret identity to screen
for superhero in superhero_list:
    print(superhero.get_given_name() + " " + superhero.get_family_name())

```

Output:

```

Bruce Wayne batman@batcave.com m
fighting crime
0

Bruce Wayne batman@batcave.com m
fighting crime
2

tony@ironman.com
superman@upupandaway.com
Clark Kent superman@upupandaway.com m
flying high
1



tony@ironman.com
Clark Kent superman@upupandaway.com m
kryptonite kaos
1

tony@ironman.com

Displaying superhero_list:
Bruce Wayne
Tony Stark
Clark Kent

```

MARKING CRITERIA

  University of South Australia		Assessment feedback	
Problem Solving and Programming (COMP 1039)			
Assignment 2 - Weighting: 15% - Due: Week 12, 2017			
NAME:	MAX MARK	MARK	COMMENT
PRODUCES CORRECT RESULTS (OUTPUT) — PART I			
length Test List length: 7 List length: 0	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
to string Test List is: r, i, n, g, i, n, g List is: r-i-n-g-i-n-g List is:	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
count Test 2 0 0	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
find Test 3 -1	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
insert value Test ['one', 'two', 'three', 'four', 'five', 'six'] ['p', 'i', 't'] ['s', 'p', 'i', 't'] ['s', 'p', 'i', 't', 's']	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
remove value Test ['one', 'two', 'three', 'four', 'five', 'six'] ['p', 'i', 't'] ['s', 'p', 'i', 't'] ['s', 'p', 'i', 't', 's']	5		<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
----- length Test List length: 6 to string Test List is: 1, 7, 2, 3, 7, 7 List is: 1 - 7 - 2 - 3 - 7 - 7 count Test 3 find Test 1 insert value Test [1, 2, 3, 4, 5, 6] remove value Test [1, 4, 5, 6]			<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
ADHERES TO SPECIFICATIONS (CODE) — PART I			
Function length(my_list) Function to_string(my_list, sep=', ' Function count(my_list, value) Function find(my_list, value) Function insert_value(my_list, value, position) Function remove_value(my_list, value, position) Well constructed loops. Appropriate if statements. No global variables.			<input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output <input type="checkbox"/> -1 No or incorrect output
Should not use the following: - Built-in functions (other than range() and str() functions)) - Slice expressions to select range of elements - String or list methods (other than list_name.append() method)) Use of list_function.py file. Use of assign2_partI_test_file.py file.			<input type="checkbox"/> -2 use of built-in functions not allowed <input type="checkbox"/> -2 use of slicing i.e. [:,-1] <input type="checkbox"/> -2 use of methods not allowed <input type="checkbox"/> -2 using break/return to exit loops
		30	

Problem Solving and Programming (COMP 1039)

Assignment 2 – Weighting: 15% - Due: Week 12, 2017

NAME:

PRODUCES CORRECT RESULTS(OUTPUT) – PART II

Please enter choice [summary|add|remove|search|update|quit]

summary

Profile Summary

Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
Tony Stark

Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
Jimmy Fallon
Fox Mulder
John Mayer
Tony Stark

MAX
MARK

MARK

COMMENT

2

- ☐ -3 No or incorrect line spacing
- ☐ -2 No or incorrect menu display
- ☐ -1 For each missing or incorrect info (up to 4 marks)
- ☐ -1 For each formatting error (up to 2 marks)

4

add

Please enter email address: homer@doh.com
Please enter given name: Homer
Please enter family name: Simpson
Please enter gender: m
Please enter current status: mmm donuts... can't talk, eating!
Successfully added homer@doh.com to profiles.

4

- ☐ -1 For each output/prompt/msg not to specs (up to 4 marks)

remove

Please enter email address: tony@ironman.com
Successfully removed tony@ironman.com from profiles.

8

- ☐ -8 If incorrect results (-4 if not removing from friends list but from profiles only)
- ☐ -1 For each output/prompt/msg not to specs (up to 4 marks)

search

Please enter email address: john@guitar.com
John Mayer (m | john@guitar.com)
- Waiting on the world to change
- Friends (2):
Katy Perry
David Guetta

4

- ☐ -1 For each output/prompt/msg not to specs (up to 4 marks)

update

Please enter email address: homer@doh.com
Update Homer Simpson [status|add friend|remove friend]: **status**
Please enter status update: doh!
Update status for Homer Simpson:
Homer Simpson (m | homer@doh.com)
- doh!
- No friends yet...

2

- ☐ -2 If incorrect results
- ☐ -1 For each output/prompt/msg not to specs (up to 1 mark)

Update Homer Simpson [status|add friend|remove friend]: **add friend**
Please enter email address of friend to add: john@guitar.com
Added John updated profile is:
Homer Simpson (m | homer@doh.com)
- doh!
- Friends (1):
John Mayer

2

- ☐ -2 If incorrect results
- ☐ -1 For each output/prompt/msg not to specs (up to 1 mark)

Update Phil Dunphy [status|add friend|remove friend]: **Remove friend**
Please enter email address of friend to remove: robbie@football.com
Removed robbie@football.com updated profile is:
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face
- Friends (1):
Fox Mulder

2

- ☐ -2 If incorrect results
- ☐ -1 For each output/prompt/msg not to specs (up to 1 mark)

Output file (new profiles.txt)

2

- ☐ -2 If output file does not exist
- ☐ -2 If incorrect results in file
- ☐ -2 If output not to specs in file

ADHERES TO SPECIFICATIONS (CODE) – PART II

While loop for menu/prompt (choice != "quit")
Appropriate control structures (in general)
Use of the following functions:

- read_file(filename, profile_list)
- write_to_file(filename, profile_list)
- display_summary(profile_list)
- find_profile(profile_list, email)
- add_profile(profile_list)
- remove_profile(profile_list)

Use of profile.py file and Profile class
Should **not** use the following: Built-in functions, Slice expressions to select range of elements, String or list methods (other than list_name.append() method), Global variables.
Validation of user input – messages:
Not a valid command – please try again.
Not a valid command – returning to main menu.

- ☐ -2 No or incorrect loop
- ☐ -2 No or incorrect control structures
- ☐ -2 No or incorrect function read_file
- ☐ -2 No or incorrect function write_to_file
- ☐ -2 No or incorrect function display_sum
- ☐ -2 No or incorrect function find_profile
- ☐ -2 No or incorrect function add_profile
- ☐ -2 No or incorrect function remove_profile
- ☐ -2 No or incorrect use of file
- ☐ -2 for each should not be using
- ☐ -2 No validation of user input
- ☐ -2 For using break/return/exit statements to exit loops

STYLE (BOTH PARTS): Comments (your details, prog. description, all variable definitions & code), meaningful variable names

5

- ☐ -4 Insufficient comments
- ☐ -4 Insufficient code layout
- ☐ -4 Non-descriptive variable names

TOTAL

65

SAMPLE OUTPUT – PART I

Sample output 1:

Start Testing!

length Test

List length: 7

List length: 0

to_string Test

List **is**: r, i, n, g, i, n, g

List **is**: r-i-n-g-i-n-g

List **is**:

count Test

2

0

0

find Test

3

-1

insert_value Test

['one', 'two', 'three', 'four', 'five', 'six']

['p', 'i', 't']

['s', 'p', 'i', 't']

['s', 'p', 'i', 't', 's']

remove_value Test

['r', 'i', 'g']

['i', 'n', 'g']

['r', 'i', 'n']

length Test

List length: 6

to_string Test

List **is**: 1, 7, 2, 3, 7, 7

List **is**: 1 - 7 - 2 - 3 - 7 - 7

count Test

3

find Test

1

```
insert_value Test  
[1, 2, 3, 4, 5, 6]
```

```
remove_value Test  
[1, 4, 5, 6]
```

```
End Testing!
```

SAMPLE OUTPUT – PART II

Sample output 1:

```
Please enter choice [summary|add|remove|search|update|quit]: asdf
Not a valid command - please try again.
Please enter choice [summary|add|remove|search|update|quit]: summary
=====
Profile Summary
=====
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
    tony@ironman.com
-----
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
    katy@perry.com
    dguetta@willworkwithanyone.org
-----
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
    john@guitar.com
    dguetta@willworkwithanyone.org
    jimmy@tonightshow.com
-----
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
    katy@perry.com
    john@guitar.com
    tony@ironman.com
    fox@findthetruth.com
    robbie@football.com
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
    robbie@football.com
```

```

    tony@ironman.com
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
    jimmy@tonightshow.com
    fox@findthetruth.com
    john@guitar.com
    tony@ironman.com
-----
=====
Please enter choice [summary|add|remove|search|update|quit]: quit
- Program terminating -
NOTE: Your program should output the following information to a file - new_profiles.txt.
Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0

Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2
robbie@football.com
fox@findthetruth.com
John Mayer john@guitar.com m
Waiting on the world to change!
2
katy@perry.com
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f
Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
5
katy@perry.com
john@guitar.com
tony@ironman.com
fox@findthetruth.com
robbie@football.com
Jimmy Fallon jimmy@tonightshow.com m
I wish I was as good as Letterman, thank goodness he's retiring.
2
robbie@football.com
tony@ironman.com
Robbie Gray robbie@football.com m

```

```
Training hard... can we win? Yes we Ken!
4
jimmy@tonightshow.com
fox@findthetruth.com
john@guitar.com
tony@ironman.com
```

Sample output 2:

```
Please enter choice [summary|add|remove|search|update|quit]: add
Please enter email address: john@guitar.com
john@guitar.com already exists in profiles.
Please enter choice [summary|add|remove|search|update|quit]: add
Please enter email address: homer@doh.com
Please enter given name: Homer
Please enter family name: Simpson
Please enter gender: m
Please enter current status: mmmmmmm.... donuts.... :)
Successfully added homer@doh.com to profiles.
Please enter choice [summary|add|remove|search|update|quit]: summary
```

```
=====
Profile Summary
=====
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
    tony@ironman.com
-----
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
    katy@perry.com
    dguetta@willworkwithanyone.org
-----
Katy Perry (f | katy@perry.com)
- Waiting on John to change.

- Friends (3):
```

```

john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
-----
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
    katy@perry.com
    john@guitar.com
    tony@ironman.com
    fox@findthetruth.com
    robbie@football.com
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
    robbie@football.com
    tony@ironman.com
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
    jimmy@tonightshow.com
    fox@findthetruth.com
    john@guitar.com
    tony@ironman.com
-----
Homer Simpson (m | homer@doh.com)
- mmmmmmm.... donuts.... :)
- No friends yet...
-----
=====

Please enter choice [summary|add|remove|search|update|quit]: quit

-- Program terminating --

NOTE: Your program should output the following information to a file - new_profiles.txt.

Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2
robbie@football.com
fox@findthetruth.com

```



```

John Mayer john@guitar.com m
Waiting on the world to change!
2
katy@perry.com
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f
Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
5
katy@perry.com
john@guitar.com
tony@ironman.com
fox@findthetruth.com
robbie@football.com
Jimmy Fallon jimmy@tonightshow.com m
I wish I was as good as Letterman, thank goodness he's retiring.
2
robbie@football.com
tony@ironman.com
Robbie Gray robbie@football.com m
Training hard... can we win? Yes we Ken!
4
jimmy@tonightshow.com
fox@findthetruth.com
john@guitar.com
tony@ironman.com
Homer Simpson homer@doh.com m
mmmmmm.... donuts.... :)
0

```

Sample output 3:

```

Please enter choice [summary|add|remove|search|update|quit]: remove

Please enter email address: bruce.wayne@batcave.com

bruce.wayne@batcave.com is not found in profiles.

Please enter choice [summary|add|remove|search|update|quit]: remove

Please enter email address: robbie@football.com

Successfully removed robbie@football.com from profiles.

Please enter choice [summary|add|remove|search|update|quit]: summary

```

```
=====
Profile Summary
=====
```

```
-----
Fox Mulder (m | fox@findthetruth.com)
```

- The truth is out there!
- Friends (1):
 tony@ironman.com

```
-----
Tony Stark (m | tony@ironman.com)
```

- Saving the world is hard work - no time for friends.
- No friends yet...

```
-----
Phil Dunphy (m | phil@dunphy.com)
```

- wtf? = why the face?
- Friends (1):
 fox@findthetruth.com

```
-----
John Mayer (m | john@guitar.com)
```

- Waiting on the world to change!
- Friends (2):
 katy@perry.com
 dguetta@willworkwithanyone.org

```
-----
Katy Perry (f | katy@perry.com)
```

- Waiting on John to change.
- Friends (3):
 john@guitar.com
 dguetta@willworkwithanyone.org
 jimmy@tonightshow.com

```
-----
David Guetta (m | dguetta@willworkwithanyone.org)
```

- Will collaborate with anyone who has a heartbeat.
- Friends (4):
 katy@perry.com
 john@guitar.com
 tony@ironman.com
 fox@findthetruth.com

```
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
```

- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (1):
 tony@ironman.com

```
=====
Please enter choice [summary|add|remove|search|update|quit]: quit
```

```
-- Program terminating --
```

NOTE: Your program should output the following information to a file - new_profiles.txt.

```

Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
1
fox@findthetruth.com
John Mayer john@guitar.com m
Waiting on the world to change!
2
katy@perry.com
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f
Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
4
katy@perry.com
john@guitar.com
tony@ironman.com
fox@findthetruth.com
Jimmy Fallon jimmy@tonightshow.com m
I wish I was as good as Letterman, thank goodness he's retiring.
1
tony@ironman.com

```

Sample output 4:

```

Please enter choice [summary|add|remove|search|update|quit]: search

Please enter email address: elvis@allshookup.org

elvis@allshookup.org is not found in profiles.

Please enter choice [summary|add|remove|search|update|quit]: search

Please enter email address: dguetta@willworkwithanyone.org

David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
  katy@perry.com

```

```
john@guitar.com m
tony@ironman.com
fox@findthetruth.com
robbie@football.com
Please enter choice [summary|add|remove|search|update|quit]: quit
```

```
-- Program terminating --
```

NOTE: Your program should output the following information to a file - new_profiles.txt.

```
Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2
robbie@football.com
fox@findthetruth.com
John Mayer john@guitar.com m
Waiting on the world to change!
2
katy@perry.com
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f
Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
5
katy@perry.com
john@guitar.com
tony@ironman.com
fox@findthetruth.com
robbie@football.com
Jimmy Fallon jimmy@tonightshow.com m
I wish I was as good as Letterman, thank goodness he's retiring.
2
robbie@football.com
tony@ironman.com
Robbie Gray robbie@football.com m
Training hard... can we win? Yes we Ken!
4
jimmy@tonightshow.com
```

fox@findthetruth.com
john@guitar.com
tony@ironman.com

Sample output 5:

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: robbie@football.com
Update Robbie Gray [status|add_friend|remove_friend]: add_friend
Please enter email address of friend to add: katy@perry.com
Added Katy updated profile is:
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (5):
Jimmy Fallon
Fox Mulder
John Mayer
Tony Stark
    katy@perry.com
Please enter choice [summary|add|remove|search|update|quit]: summary
```

```
=====
Profile Summary
=====
```

```
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
    tony@ironman.com
-----
```

```
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
```

```
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
```

```
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
    katy@perry.com
    guetta@willworkwithanyone.org
-----
```

```
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
    john@guitar.com m
```

guetta@willworkwithanyone.org
immy@tonightshow.com

David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
 katy@perry.com
 john@guitar.com m
 tony@ironman.com
 fox@findthetruth.com
 robbie@football.com

Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
 robbie@football.com
 tony@ironman.com

Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (5):
 immy@tonightshow.com
 fox@findthetruth.com
 john@guitar.com m
 tony@ironman.com
 katy@perry.com

=====

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: tony@ironman.com

Update Tony Stark [status|add_friend|remove_friend]: add_friend

Please enter email address of friend to add: homer@doh.com
homer@doh.com is not found in profiles.

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: robbie@football.com

Update Robbie Gray [status|add_friend|remove_friend]: add_friend

Please enter email address of friend to add: john@guitar.com

John is already a friend.

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: dana.scully@truth.com

dana.scully@truth.com is not found in profiles.

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: robbie@football.com

Update Robbie Gray [status|add_friend|remove_friend]: add

Not a valid command - returning to main menu.

Please enter choice [summary|add|remove|search|update|quit]: summary

```
=====
Profile Summary
=====
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
  tony@ironman.com
-----
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
Robbie Gray
  fox@findthetruth.com
-----
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (2):
  katy@perry.com
  guetta@willworkwithanyone.org
-----
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
  john@guitar.com m
  guetta@willworkwithanyone.org
  inmy@tonightshow.com
-----
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
  katy@perry.com
  john@guitar.com m
  tony@ironman.com
  fox@findthetruth.com
```

```

    robbie@football.com
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
    robbie@football.com
Tony Stark
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (5):
    jimmy@tonightshow.com
    fox@findthetruth.com
    john@guitar.com m
    tony@ironman.com
    katy@perry.com
-----
=====

Please enter choice [summary|add|remove|search|update|quit]: quit

-- Program terminating --

NOTE: Your program should output the following information to a file - new_profiles.txt.
Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2
robbie@football.com
fox@findthetruth.com
John Mayer john@guitar.com m
Waiting on the world to change!
2
katy@perry.com dguetta@willworkwithanyone.org Katy Perry katy@perry.com f Waiting on
John to change. 3 john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
5
katy@perry.com
john@guitar.com
tony@ironman.com
fox@findthetruth.com
robbie@football.com

```


Jimmy Fallon jimmy@tonightshow.com m
 I wish I was as good as Letterman, thank goodness he's retiring.
 2
 robbie@football.com

 tony@ironman.com
 Robbie Gray robbie@football.com m
 Training hard... can we win? Yes we Ken!
 5
 jimmy@tonightshow.com
 fox@findthetruth.com
 john@guitar.com
 tony@ironman.com
 katy@perry.com

Sample output 6:

```

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: fox@findthetruth.com

Update Fox Mulder [status|add_friend|remove_friend]: remove_friend

Please enter email address of friend to remove: lenny.kravitz@rock.com
lenny.kravitz@rock.com is not Fox's friend.

Please enter choice [summary|add|remove|search|update|quit]: update

Please enter email address: john@guitar.com
Update John Mayer [status|add_friend|remove_friend]: remove_friend

Please enter email address of friend to remove: katy@perry.com
Removed katy@perry.com updated profile is:

John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (1):
    dguetta@willworkwithanyone.org

Please enter choice [summary|add|remove|search|update|quit]: summary

=====
Profile Summary
=====
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
    tony@ironman.com
-----
  
```

```

Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
John Mayer (m | john@guitar.com)
- Waiting on the world to change!
- Friends (1):
    dguetta@willworkwithanyone.org
-----
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
    john@guitar.com m
    guetta@willworkwithanyone.org
Jimmy Fallon
-----
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
- Friends (5):
    katy@perry.com
    john@guitar.com m
    tony@ironman.com
    fox@findthetruth.com
    robbie@football.com
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
    jimmy@tonightshow.com
    fox@findthetruth.com
    john@guitar.com m
    tony@ironman.com
-----
=====

Please enter choice [summary|add|remove|search|update|quit]: quit

-- Program terminating --

```

NOTE: Your program should output the following information to a file - new_profiles.txt.

Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2

robbie@football.com
fox@findthetruth.com
John Mayer john@guitar.com m
Waiting on the world to change!
1
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f
Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org
jimmy@tonightshow.com
David Guetta dguetta@willworkwithanyone.org m
Will collaborate with anyone who has a heartbeat.
5
katy@perry.com

john@guitar.com
tony@ironman.com
fox@findthetruth.com

robbie@football.com
Jimmy Fallon jimmy@tonightshow.com m
I wish I was as good as Letterman, thank goodness he's retiring.
2
robbie@football.com
tony@ironman.com
Robbie Gray robbie@football.com m
Training hard... can we win? Yes we Ken!
4
jimmy@tonightshow.com
fox@findthetruth.com
john@guitar.com
tony@ironman.com

Sample output 7:

```
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: brad@pitt.com
brad@pitt.com is not found in profiles.
Please enter choice [summary|add|remove|search|update|quit]: update
Please enter email address: john@guitar.com
Update John Mayer [status|add_friend|remove_friend]: status
Please enter status update: Slow dancing in a burning room...
Updated status for John Mayer:
John Mayer (m | john@guitar.com)
- Slow dancing in a burning room...
- Friends (2):
    katy@perry.com
David Guetta
```

```
Please enter choice [summary|add|remove|search|update|quit]: summary
```

```
=====
Profile Summary
=====
```

```
-----
Fox Mulder (m | fox@findthetruth.com)
- The truth is out there!
- Friends (1):
    tony@ironman.com
-----
```

```
Tony Stark (m | tony@ironman.com)
- Saving the world is hard work - no time for friends.
- No friends yet...
-----
```

```
Phil Dunphy (m | phil@dunphy.com)
- wtf? = why the face?
- Friends (2):
    robbie@football.com
    fox@findthetruth.com
-----
```

```
John Mayer (m | john@guitar.com)
- Slow dancing in a burning room...
- Friends (2):
    katy@perry.com
    dguetta@willworkwithanyone.org
-----
```

```
Katy Perry (f | katy@perry.com)
- Waiting on John to change.
- Friends (3):
    John Mayer
    David Guetta
    Jimmy Fallon
-----
```

```
David Guetta (m | dguetta@willworkwithanyone.org)
- Will collaborate with anyone who has a heartbeat.
```

```

- Friends (5):
    katy@perry.com
    john@guitar.com m
    tony@ironman.com
    fox@findthetruth.com
    robbie@football.com
-----
Jimmy Fallon (m | jimmy@tonightshow.com)
- I wish I was as good as Letterman, thank goodness he's retiring.
- Friends (2):
    robbie@football.com
    tony@ironman.com
-----
Robbie Gray (m | robbie@football.com)
- Training hard... can we win? Yes we Ken!
- Friends (4):
    jimmy@tonightshow.com
    fox@findthetruth.com
    john@guitar.com m
    tony@ironman.com
-----
=====

Please enter choice [summary|add|remove|search|update|quit]: quit

-- Program terminating --

NOTE: Your program should output the following information to a file - new_profiles.txt.
Fox Mulder fox@findthetruth.com m
The truth is out there!
1
tony@ironman.com
Tony Stark tony@ironman.com m
Saving the world is hard work - no time for friends.
0
Phil Dunphy phil@dunphy.com m
wtf? = why the face?
2
robbie@football.com
fox@findthetruth.com
John Mayer john@guitar.com m
Slow dancing in a burning room...
2
katy@perry.com
dguetta@willworkwithanyone.org
Katy Perry katy@perry.com f

Waiting on John to change.
3
john@guitar.com
dguetta@willworkwithanyone.org

```

jimmy@tonightshow.com

David Guetta dguetta@willworkwithanyone.org m

Will collaborate with anyone who has a heartbeat.

5

katy@perry.com

john@guitar.com

tony@ironman.com

fox@findthetruth.com

robbie@football.com

Jimmy Fallon jimmy@tonightshow.com m

I wish I was as good as Letterman, thank goodness he's retiring.

2

robbie@football.com

tony@ironman.com

Robbie Gray robbie@football.com m

Training hard... can we win? Yes we Ken!

4

jimmy@tonightshow.com

fox@findthetruth.com

john@guitar.com

tony@ironman.com