Assignment 3

Key information (Front page)

How will I be assessed?

60% team mark (common): based on submission

- Approximately 20% of the team mark from test cases.
- Approximately 80% of the team mark from review by your teacher as per rubric (in a few slides).

35% based on interview (individual)

Details To Be Announced.

5% for submitting feedback to team members (individual)

Details To Be Announced.

5% bonus marks for FIT1045 students completing part 8

This means that the 60% of the team mark can be made up for (in part) by up to 5% if you successfully complete part 8. For example, if you were to obtain 57 of the 60% for the team mark, but have correctly solved part 8, then your team mark becomes 60%. If it was 50, you' get 55% (if part 8 is perfect).

Review of your work by your tutor

Your work will be looked at **overall** (not necessarily every single task) and this will align with the rubric included.

This is to find evidence you are able to use the different techniques you've learnt about so far in the unit (e.g. do you understand how to meaningfully design a conditional to achieve some broader goal)

Maximising your marks from tutor review

You should aim to ensure your program includes the elements in the rubric (next page).

To ensure this is the best representation of your ability, you should ensure you contribute to tasks in workshops and applied classes so that the **feedback** your group receives will be **meaningful** for you and will guide you in improving proficiency.

Teams smaller than 4 (change 2)

Some of you ended up in teams smaller than 4.

- The A3 team google sheet as students unenroll from the unit. If you believe it is not up to date, please let us know via a private post on ed.
- If you do not have a team of 4 **according to the google sheet**, then you have less work to do:
    - If you are in FIT1045, then you do not have to do Task 7.
    - If you are in FIT1053, then you do not have to do Task 8.
- If your team has 2 or fewer team members, then please reach out to us so we can try to put you in a team of 2.

Academic integrity

How will this be checked?

Your code will be compared against every other students' work in the unit.

You should also assume that anything you are able to google can be easily found by the teaching team and compared against your work.

You will also be **individually** interviewed on your understanding of different components of the program, whatever your contribution to a particular component might be. You are expected to understand all parts of the program equally well. Make sure you work as a team to ensure that.

How can I avoid academic integrity issues?

- Never copy code from anywhere. If you learn something useful online, rewrite it from scratch. It's also the best way to make sure you have understood it. If you're concerned you may cause an academic integrity case by copying something on the internet, the easiest way to avoid this is to not search anything too specific. Once you read a solution, it is very hard to forget it.
- If a fellow student asks you for a solution to a question, try instead to figure out what it is that they do not understand about the unit's content that prevents them from finding the solution themselves. Give a man a fish, and you feed him for a day. Teach a man to fish, and you feed him for a lifetime. Also, remember that giving your solution is just as much of an Academic Integrity breach as receiving it!
- You may find yourself in a situation where you feel like you physically cannot submit the assignment on time. Remember that you can submit an extension request (see Moodle AU or Moodle MA), and you can seek help (see Moodle AU or Moodle MA). If nothing works, remember that failures are part of the learning journey. And what is more important to you, an assignment mark or acting honourably?

Workspace and Automated Testing

We have included some test files for each of the Python files we expect you to produce (in the Assignment 3 scaffold workspace, see details below).

To run these tests yourself you will need to go to the console and run

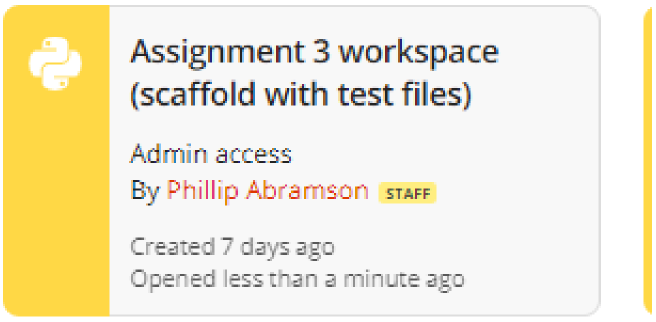
```
python test_XXX.py
```

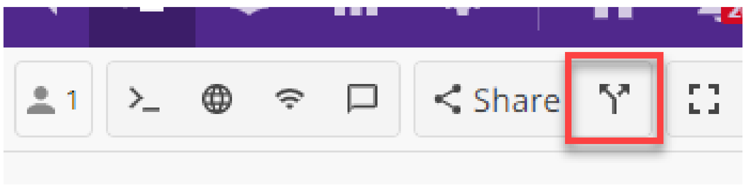
The results you see in the console will be the results of running these tests

We will use the test cases for the 20% of your marks coming from automated tests; do not modify these files as they will simply be overwritten by your TA when it comes to marking your work.

Accessing the workspace

In this assignment, you are expected to work in teams of 4. You will need to meet regularly with your team and collaborate using the workspace in Ed. There will be a scaffold workspace in Ed for you to fork using the button in the top right as shown below.


Assignment 3 workspace (scaffold with test files)
Admin access
By Phillip Abramson STAFF
Created 7 days ago
Opened less than a minute ago



Make sure to share your workspace with you teammates. We recommend you use a single workspace, but you can use more if that works better for your team.

Once you know the tutor who will mark you, add them to your workspace prior to the interview. They will be able to see who has contributed over the course of the assessment period through the replay option.

After you submit the assignment, you will be asked to complete a peer review of one another's contributions to the team, the submission of which will contribute to your individual mark.

Checking that you have the right test files and that they are not modified

One way to check that you are using the right test files is to type the command `md5sum test_*` in the terminal of your workspace. If you have the correct test files, you should see exactly this:

```
b168cac4004e16c4dfdfbf24976de910  test_social_network_class_based.py
a9bf2eebdacfc960d59755d2b5585b94  test_social_network_class_with_followers.py
9509ae663e8de5525ce27320f1148905  test_social_network_dictionary_based.py
```

(updated as part of change 3).

If you don't, try to grab a fresh copy of the test files from the workspace and try again. Please let us know if that does not work.

Assignment Rubric

Submission information

When you have completed your work, download it from your workspace and upload it on the Moodle "Assignment 3 submission" box on the assessment page (Moodle AU - Moodle MA). One submission per team!

The date of submission on Moodle is used to determine late penalties.

i You can download all files in the workspace as a zip file by right-clicking anywhere in the file sidebar and selecting "Download All".

Important: Dos and Don’ts

Do

- ✓ Add your teammates to your ed workspace.
- ✓ Add the TA marking your assignment to your ed workspace once you know who they will be.
- ✓ Run your work regularly and test with different scenarios.
- ✓ Use the class, function names and arguments given when writing your code.
- ✓ Program, review and discuss your code with your teammates.

Don’t

- ✗ Change the test files provided. We will use our own.
- ✗ Work or ask other members of your team to work in isolation.
- ✗ Import any libraries or packages aside from datetime (you won’t need them).
    - you may otherwise find it difficult to demonstrate some aspects in the rubric or explain how components of the program function

Changelog (3 changes)

Changes made to the instructions will be reported here. All times are Melbourne time.

Change 3 – 27/04/2022 at 10:30

Released task 8: FIT1053 + Bonus marks component.

Please be sure to use the latest version of the test file `test_social_network_class_with_followers.py`. md5 sums of the test files are

```
b168cac4004e16c4dfdfbf24976de910  test_social_network_class_based.py
a9bf2eebdacfc960d59755d2b5585b94  test_social_network_class_with_followers.py
9509ae663e8de5525ce27320f1148905  test_social_network_dictionary_based.py
```

Change 2 - 21/04/2022 at 09:20

Updated this slide on what happens to teams smaller than 4.

Change 1 - 17/04/2022 at 11:50

Uploaded Rubric.

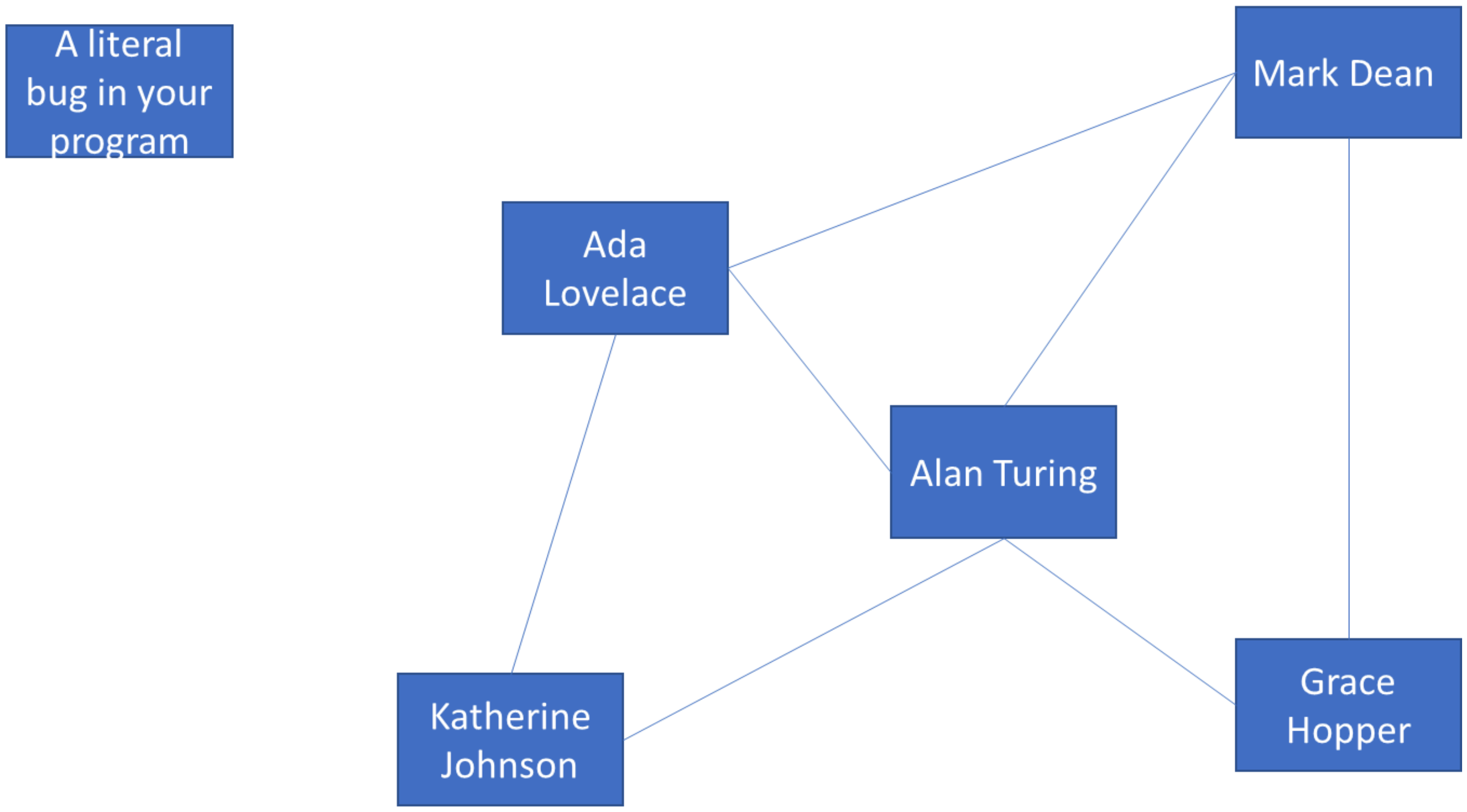
Overview of Assignment

This assignment will have you and your friends creating a structure and program to support a social network. This will of course be quite a **limited** system (full systems of this nature require a firmer grasp of a wide set of data structures and algorithms together with privacy and security implications).

The intention for our version is to represent the people in the network, the connections between them and the posts they make and ultimately to allow us to make birthday post messages for our friends.

As such we will call this system BFFday (pronounced buh-ff-day)

Sample network



In the image above, we see a number of computer science figures connected by lines representing friendships (noting of course that some of these friendships are impossible; for instance, Alan Turing has tragically passed several years before Mark Dean was even born).

In the diagram above we have Ada Lovelace (the first computer programmer), Mark Dean (one of the key figures involved in developing the personal computer), Alan Turing (widely considered the father of computer science), Katherine Johnson (a human computer critical to NASA's early human space flights) and Grace Hopper (inventor of one of the first high level computer languages COBOL). Ada has a connection to Katherine, Mark and Alan (meaning they are friends), Alan and Katherine are also friends, Mark and Alan are also friends and finally Grace is friends with Alan and Mark.

We also have another "individual" in the network which is "a literal bug in your program". Of course, none of us would be keen to have that as friend, which is why they are not connected to any of our prestigious computer scientists.

Your task

By the completion of this assignment you should have a class structure representing the set of people and posts as well as one for individuals to hold their information and connections to others.

Given you will only be learning about classes in week 8, we will first prepare this program as a sequence of independent functions which pass around dictionaries (hence without the use of classes) and will later revise it to suit.

Key steps

- Take input to create individuals, form and break connections, and check for connections.
- Holding a structure with all people in the network that we can add to and search.
- Taking string input and turning it into a network.
- Creating a post from a particular user's account which may or may not reference their friends.
- Identify who has a birthday coming up and make a post for them.
- Revising the entire program to work with classes instead of transactional style functions.
- Represent a year's worth of birthday posts being made in the network.

1. Representing a person as a Python dict

Your task

Prepare a file called `social_network_dictionary_based.py` which **contains** (at least) each of the functions defined below. Ensure that your function names match the ones we provide, as the automated marking will use these names for testing.

You will need to use

The `date` object within Python's datetime library <https://docs.python.org/3/library/datetime.html>

Function list

`make_person(this_id,name,date_of_birth)` details

▶ Expand

`find_friendX_inY(person_X,person_Y)` details

▶ Expand

`make_friendship(person_X,person_Y)` details

▶ Expand

`end_friendship(person_X,person_Y)` details

▶ Expand

`birthday_within_X_days_of_Y(person,days,comparison_date)` details

▶ Expand

2. Representing a dictionary of people

Your task

Add to the file `social_network_dictionary_based.py` the following functions. Ensure that your function names match the ones we provide, as the automated marking will use these names for testing.

You will need to use

your code from part 1

Function list

`add_person(dict_of_people,name,date_of_birth)` details

▶ Expand

`get_person_by_id(dict_of_people,find_id)` details

▶ Expand

3. Creating a network from input

Your task

Add to the file **`social_network_dictionary_based.py`** the function `convert_lines_to_friendships(lines)` as defined below. Ensure that the function name matches, as the automated marking will use these names for testing.

You should also add a main-program-only section to the file (e.g. as below)

```
if __name__ == "__main__":
    #run your code here
```

where the `convert_lines_to_friendships(lines)` function can be triggered and given input from the user

Supporting information

For the purposes of this program, you can assume a given social network can be represented as a sequence of people or pairs of people (which represents a friendship)

The input format is a sequence of strings with each string being in one of the following two formats

- "*name*`,`*date_of_birth*<->*name2*`,`*date_of_birth2*" **OR**
- "*name*`,`*date_of_birth*"

In this case, the dash character (`<->`) represents a friendship (a lack of one means we're just adding a person without a friendship). Dates are represented in the format of YEAR`-`MONTH`-`DAY

e.g.

```
'Fred,2022-02-01<->Jenny,2004-11-18'
'Jiang,1942-09-16<->Sasha,1834-02-02'
'Corey,2015-05-22'
'Sasha,1834-02-02<->Amir,1981-08-11'
```

represents that Fred (1st Feb 2022) is friends with Jenny (18th Nov 2004), Jiang (16th Sep 1942) is friends with Sasha (2nd Feb 1834 [yes they are an industrial revolution era vampire, but that's not relevant for this assignment]), Corey (22nd May 2015) doesn't (yet) have friends, and Sasha is also friends with Amir (11th Aug 1981)

You will need to use

your code from parts 1 and 2

`convert_lines_to_friendships(lines)` details

Purpose:

- Read a list of strings defined in the supporting information
- Create a dictionary of person dictionary objects as defined in part 1.
    - this dictionary should assign IDs to each person in the order they are added (and reference them by ID)
        - refer to part 2, **`add_person(dict_of_people,name,date_of_birth)`** details
    - this dictionary should contain **unique** people only
        - you can assume no one shares a name (though birthdays are not necessarily unique)
        - this means if a person appears more than once then we are creating a friendship with an existing person in our dictionary
- returns the full dictionary once complete

You should assume the string is always given in the format expected

Arguments:

- `lines` – a list of strings in the format given in the supporting information

Return Value:

- The `dict` holding these properties

Sample input and output:

- **Function call 1**

```
>>people = [
        'Fred,2022-02-01<->Jenny,2004-11-18',
        'Jiang,1942-09-16<->Sasha,1834-02-02',
        'Corey,2015-05-22',
        'Sasha,1834-02-02<->Amir,1981-08-11'
]
>>convert_lines_to_friendships(people)
```

- **Return value 1:**

```
{
    1: {'friends': [2], 'history': [], 'id': 1, 'name': 'Fred', 'date_of_birth': datetime.date(2022
    2: {'friends': [1], 'history': [], 'id': 2, 'name': 'Jenny', 'date_of_birth': datetime.date(200
    3: {'friends': [4], 'history': [], 'id': 3, 'name': 'Jiang', 'date_of_birth': datetime.date(194
    4: {'friends': [3, 6], 'history': [], 'id': 4, 'name': 'Sasha', 'date_of_birth': datetime.date(
    5: {'friends': [], 'history': [], 'id': 5, 'name': 'Corey', 'date_of_birth': datetime.date(2015
    6: {'friends': [4], 'history': [], 'id': 6, 'name': 'Amir', 'date_of_birth': datetime.date(1981
}
```

- **Function call 2**

```
>>people = [
        'Carcinisation:one of the many attempts of Nature to evolve a crab,1916-12-31',
        'Recursion,1888-01-01',
        'Crabception: Our destiny: a crab made entirely of crabs,2450-10-17<->Recursion,1888-01-01'
        'Crabception: Our destiny: a crab made entirely of crabs,2450-10-17<->Carcinisation:one of 
]
>>convert_lines_to_friendships(people)
```

- **Return value 2:**

```
{
    1: {'friends': [3], 'history': [], 'id': 1, 'name': 'Carcinisation:one of the many attempts of 
    2: {'friends': [3], 'history': [], 'id': 2, 'name': 'Recursion', 'date_of_birth': datetime.date
    3: {'friends': [2, 1], 'history': [], 'id': 3, 'name': 'Crabception: Our destiny: a crab made e
}
```

4. Making a post

Your task

Add to the file `social_network_dictionary_based.py` the function `new_post(content,author,tagged)` (defined below).

Ensure that your function names match the ones we provide, as the automated marking will use these names for testing.

Supporting information

Another necessary part of our social network is a way of representing a post (a message that a particular person shared with their friends). These posts can be used to tag friends of the author (people who are not friends with the author cannot be tagged)

For the moment, we will represent a social media post as a tuple including the post contents, the author of the post and any of their friends tagged in it.

For example

```
("Hey all, spending some time with my best buds! What a day!!!", 47, [1,54,77,6])
```

would represent a post with this content

> Hey all, spending some time with my best buds! What a day!!!

which was written by the person with ID 47 and includes a tag of the following friends' IDs: 1, 54, 77, and 6.

This of course assumes that person 47 **is** friends with 1, 54, 77, and 6.

If instead 47 was **only** friends with 54 and 77 then despite *attempting* to tag 1, 54, 77, and 6, the resultant post would be

```
("Hey all, spending some time with my best buds! What a day!!!", 47, [54,77])
```

After the post is created, it will appear in the history list for the person with the ID 47.

You will need to use

Your code from parts 1–3.

`new_post(content,owner,tagged)` details

▶ Expand

5. Birthday posts

Your task

Add to the file `social_network_dictionary_based.py` the functions

- `birthdays_within_a_week_of(person_id,people_dict,comparison)`
- `make_birthday_posts(from_person_id,for_people_ids)`

(both defined below).

Ensure that your function names match the ones we provide, as the automated marking will use these names for testing.

You will need to use

- your code from parts 1–4

`birthdays_within_a_week_of(person_id,people_dict,comparison)` details

▶ Expand

`make_birthday_posts(people_dict,from_person_id,for_people_ids)` details

▶ Expand

6. Rewrite friend and people lists as classes

Your task

- Prepare a file called **`social_network_class_based.py`** which **contains** (at least) the following classes as defined below.
- Take the functions you previously wrote and include revised versions as part of these classes (as per the mapping below).
    - for any functions not explicitly mapped in the below, you should use your best judgement to place them somewhere appropriate (assuming they are needed)
- Create an `if __name__=="__main__":` block in the file which collects user input and instantiates the people class to represent a particular social network

Ensure that your class and function names match the ones we provide, as the automated marking will use these names for testing. You can add additional property and methods to the classes as needed to simplify or support your code

You will need

Your code to parts 1–5

Classes and mapping functions to them

`Person` class

▶ Expand

`SocialNetwork` class

▶ Expand

[not assessed] Your thoughts on these approaches

Question 1

Now that you've represented using both dictionaries and class instances, which do you feel makes the most sense in this setting?

- ○ dictionaries
- ○ classes

Question 2

How do you see yourself applying the lessons from this assignment in the future?

No response

7. Run through a year of birthday messages

Your task

This task will **not** include automated testing. However, if your program has passed all prior tests it is very likely you won't encounter too many bugs writing this part.

Add to your file `social_network_class_based.py` some code to simulate running through a year's worth of birthday posts.

- Receive a social network structure as input (using whatever approach you used earlier).
- Use the current year (hint: see the `datetime` library) as part of the comparison date.
- Run through every single day of this year.
- For each of these days, have each `Person` in the network check for and make (as appropriate) birthday posts for each of their friends with a birthday falling on that exact day.
- Ensure you do **not** create duplicate posts (multiple people can make a birthday post for the same friend but an individual person should not make say a dozen posts wishing the same person a happy birthday).
- Finally, print off all the posts in the `SocialNetwork` instance in the order they appear
    - You should find the posts appear in birthday order with multiple people making a post for the same (shared) friend

Hint: you may need a slightly revised version of your `make_birthday_posts` method, but be careful to limit repeated code as much as possible

Sample network

```
1 (Fred, 2022-02-01) --> 2
2 (Jenny, 2004-11-18) --> 1
3 (Jiang, 1942-09-16) --> 4
4 (Sasha, 1234-02-02) --> 3, 6
5 (Corey, 2015-05-22) -->
6 (Amir, 1981-08-11) --> 4
```

Sample posts generated (in order)

```
('Happy birthday Fred! Hope you have a good one!', 2, [1])
('Happy birthday Sasha! Hope you have a good one!', 3, [4])
('Happy birthday Sasha! Hope you have a good one!', 6, [4])
('Happy birthday Amir! Hope you have a good one!', 4, [6])
('Happy birthday Jiang! Hope you have a good one!', 4, [3])
('Happy birthday Jenny! Hope you have a good one!', 1, [2])
```

For birthday posts for the same friend from different people the order is technically arbitrary, however in our program they are made in ascending author ID order.

8. FIT1053 + Bonus marks component – threaded posts and followers

Who is this component for?

- FIT1053 students are expected to complete this as part of their assignment
- FIT1045 students who are interested in up to 5% bonus marks

Your task

Prepare a file called `social_network_class_with_followers.py` which implements the class-based social network with the following revisions:

1. People in the network should be able to **follow** one another in addition to forming friendships (people can be both followers and friends in this setup).
2. People in the network should be able to create **`threaded`** posts (posts which can have other posts as children).
    - in this case, **every** post in the network will be threaded

This program should still have an `if __name__=="__main__":` block in the file which collects user input and instantiates the people class to represent a particular social network

Hint: You can import the classes and functions from the original class-based python file and override, replace or add additional methods needed on top of this.

You will need

Your code to parts 1–6

Supporting information

Revised post representation

▶ Expand

Revised input / display format

▶ Expand

New functionality definitions

1. Allowing followers

▶ Expand

2. Create threaded posts

Create the following functions:

- **`make_threaded_post(self,content,tagged,is_private)`** -- a method of your `Person` class (or equivalent)
- **`add_child(threaded_post, content, new_post_owner, tagged, is_private)`** -- a free standing function in your **`social_network_class_with_followers.py`** file

both of these are described below

2A. **`make_threaded_post(self,content,tagged,is_private)`** definition

▶ Expand

2B. **`add_child(threaded_post, content, new_post_owner, tagged, is_private)`** definition

▶ Expand