ISAD1000 - Final Assessment

Freddy Khant, 20618166, Practical Class - Tue 4pm

Introduction

The scope of this project was to implement a simple software which performs various forms of conversions; conversions of strings and measurements. Utilising version control, it was also required that the software be tested with appropriate testing fixtures. In addition the software also had to adhere to good modularity principles and consider possible ethical and professional issues.

The functionalities chosen to be implemented were all of that of **Category 1** - string conversions, and **Category 2** - time measurement conversions

Module Descriptions

Pre-Refactoring

All functionalities to be implemented will be separated into modules that handle different forms of conversions. There will be a module for each functionality of *Category 1* and one from *Category 2*. Each of these modules will be:

- Program converting a string to upper/lower case
- Program identifying whether numeric values are in given string
- Program identifying whether a given string is a valid number
- Program removing numeric values in a given string and converting them to either uppercase or lowercase.

Post-Refactoring

All functionalities to be implemented were separated into modules that handle different forms of conversions. There is a module for each functionality of *Category 1* and one from *Category 2*. Each of these modules contain different versions of the original function, to handle input and output differently. The modules are as specified:

Strings

- Program converting a string to upper/lower case
- Program identifying whether numeric values are in given string
- Program identifying whether a given string is a valid number
- Program removing numeric values in a given string and converting them to either uppercase or lowercase.

Time

- Program converting hours to minutes
- Program converting minutes to hours
- Program converting minutes to seconds
- Program converting seconds to minutes

Each of these modules will contain:

- The initial and most basic version of the function, taking a parameter input and returning an output.

- Another version of the function that now enables the user to input a string via keyboard, giving a console output
- The last version of the function which allows input data to be read from a text file, and the output results to be written to that text file.

Modularity

How To Run

Since the programs were coded using Python:

```
python3 program.py
```

Modularity Checklist

To achieve low coupling:

- Avoid use of global variables
- Avoid use of too many parameters
- Avoid too many function calls

To achieve high cohesion:

- Make sure module does one well-defined task
- Minimize one module doing too many tasks

2/3 of the coupling checklist checked - avoided use of global variables, and excessive use of parameters 3/3 of cohesion checklist checked - each module does one well defined task, and modules are cohesive

Overall the program adheres basic modularity guidelines by:

- Avoiding low coupling - avoiding use of global variables, excessive function parameters, etc. And
- Achieving high cohesion - writing modules that do well defined task, have individuality and are cohesive

In addition, the software seeks to avoid redundancy by minimising the repetition and duplication of code. To achieve this, code is recycled and reused wherever possible.

Blackbox Test Cases

Strings

to_upper

Category	Test Data	Expected Result
lowercase	"khant"	"KHANT"
uppercase	"KHANT"	"KHANT"

Category	Test Data	Expected Result
numeric	"8166"	"8166"

to_lower

Category	Test Data	Expected Result
lowercase	"khant"	"khant"
uppercase	"KHANT"	"khant"
numeric	"8166"	"8166"

is_numeric

Category	Test Data	Expected Result
numeric	"8166"	True
partially numeric	"81KH"	True
!numeric	"8166"	False

is_number

Category	Test Data	Expected Result
numeric	"8166"	True
partially numeric	"81KH"	False
!numeric	"8166"	False

convert_upper

Category	Test Data	Expected Result
number + lowercase	“81kh”	"KH”
number + uppercase	"81KH"	"KH”
lowercase	“kh”	“KH”
uppercase	“KH”	“KH”
number	"8166"	"8166"

convert_lower

Category	Test Data	Expected Result
number + lowercase	“81kh”	"kh”
number + uppercase	"81KH"	"kh”

Category	Test Data	Expected Result
lowercase	“kh”	“kh”
uppercase	“KH”	“kh"
number	"8166"	"8166"

Time

hours_to_mins

Category	Test Data	Expected Result
n > 0	8166	489960
n < 0	-8166	"Invalid time"

mins_to_hours

Category	Test Data	Expected Result
n > 0	8166	136.1
n < 0	-8166	"Invalid time"

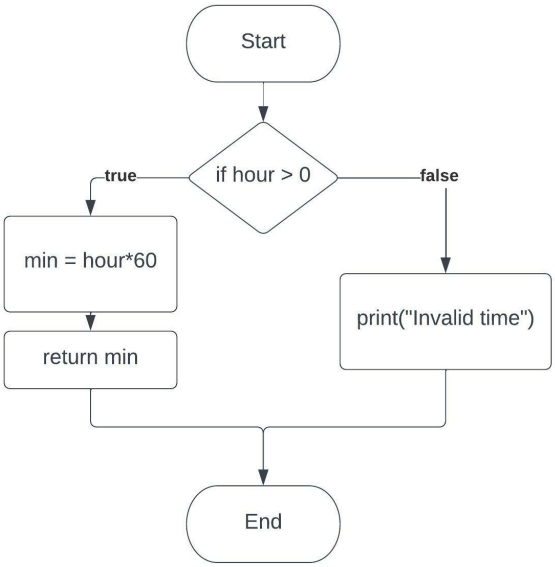
mins_to_seconds

Category	Test Data	Expected Result
n > 0	8166	489960
n < 0	-8166	"Invalid time"

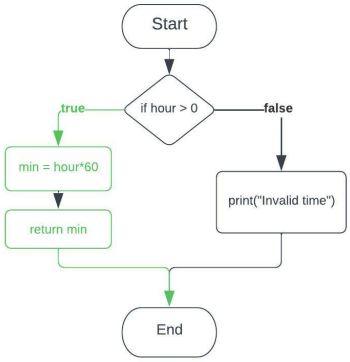
seconds_to_mins

Category	Test Data	Expected Result
n > 0	8166	136.1
n < 0	-8166	"Invalid time"

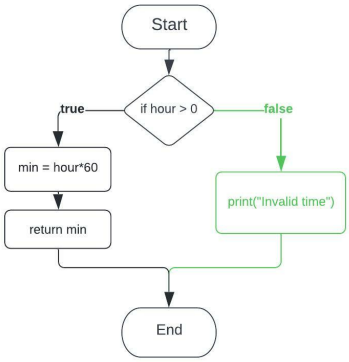
Whitebox Test Cases

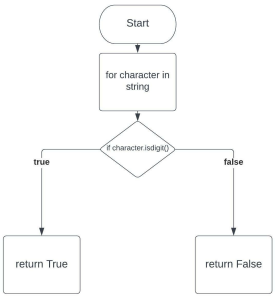


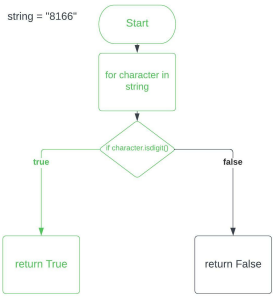
hour > 0:

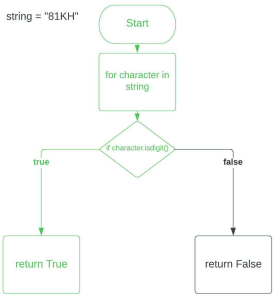


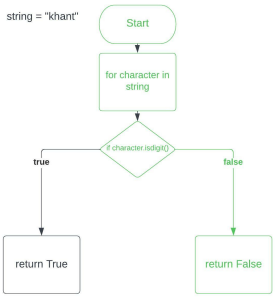
hour < 0:









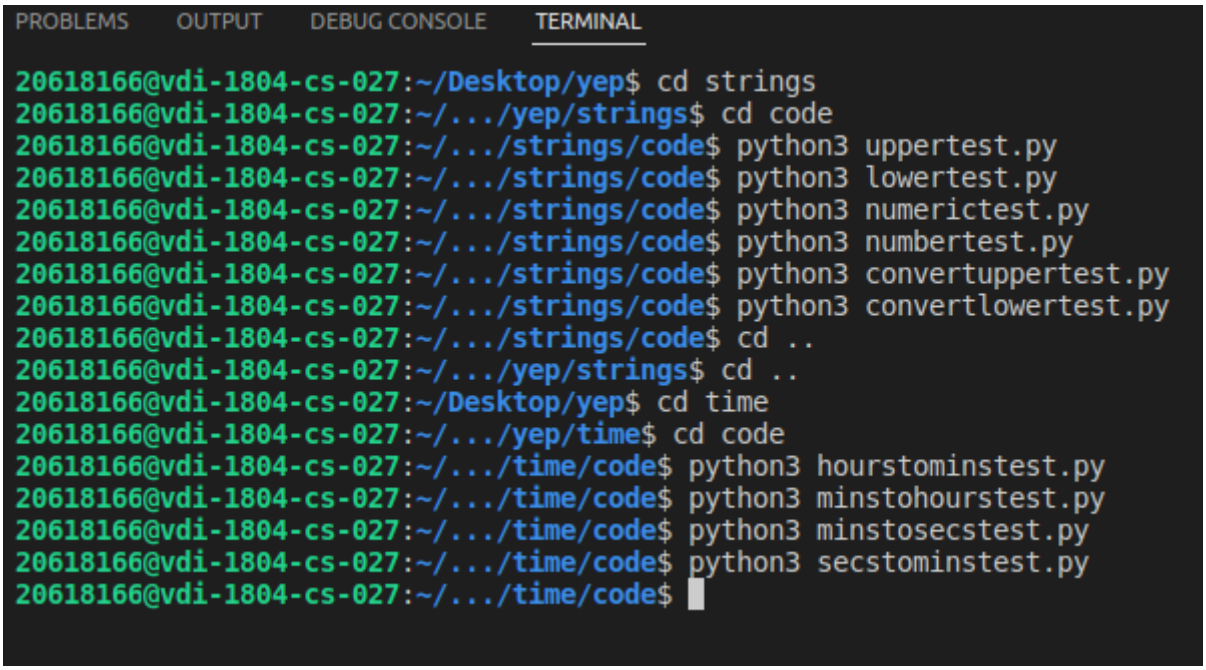


Test Implementation and Execution

How to Run

```
python3 test.py
```

Test Results



All tests passed, modules working as expected

Test Fixtures Table

Module Name	EP test design	BVA test design	WB test design	EP test code	BVA test code	WB test code
to_upper	done	not done	not done	done	not done	not done
to _lower	done	not done	not done	done	not done	not done
is_numeric	done	not done	done	done	not done	not done
is_number	done	not done	not done	done	not done	not done
convert_upper	done	not done	not done	done	not done	not done
convert_lower	done	not done	not done	done	not done	not done
hours_to_mins	done	not done	done	done	not done	not done
mins_to_hours	done	not done	not done	done	not done	not done
mins_to_seconds	done	not done	not done	done	not done	not done
seconds_to_mins	done	not done	not done	done	not done	not done

Version Control

```
20618166@vdi-1804-cs-027:.../FINAL/code$ git log --branches --graph
* commit 7cbabd7e9c0926e199f60ed926befec5d8c8bc0f (HEAD -> master, testing)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:35:54 2022 +0800
|
|     All blackbox test fixtures for time conversion modules
|
* commit 671cb0e17206803876e98bddc722fc27d6869c5d
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:34:44 2022 +0800
|
|     All test black box test fixtures for strings modules
|
* commit fcf0b2f3aca1b4f0c0aaf6f20a0dff78f73c9dc2 (time)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:33:10 2022 +0800
|
|     Modules for time including different input/output handling
|
* commit a65328932956263a961f2afe0bf5e196d59fdb1f (strings)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:30:51 2022 +0800
|
|     All modules for strings including different functionalities for input outp:...skipping...
* commit 7cbabd7e9c0926e199f60ed926befec5d8c8bc0f (HEAD -> master, testing)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:35:54 2022 +0800
|
|     All blackbox test fixtures for time conversion modules
|
* commit 671cb0e17206803876e98bddc722fc27d6869c5d
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:34:44 2022 +0800
|
|     All test black box test fixtures for strings modules
|
* commit fcf0b2f3aca1b4f0c0aaf6f20a0dff78f73c9dc2 (time)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:33:10 2022 +0800
|
|     Modules for time including different input/output handling
|
* commit a65328932956263a961f2afe0bf5e196d59fdb1f (strings)
| Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
| Date:   Tue May 31 19:30:51 2022 +0800
|
|     All modules for strings including different functionalities for input output handling
|
* commit cc33ba0e090377c471b4c8baef4a1d0d90bc5184
  Author: Freddy Khant <freddy.khant@student.curtin.edu.au>
  Date:   Tue May 31 19:29:29 2022 +0800

      Initial version of convertupper
```

Version control using git was specifically utilized to log the different versions of the modules. Initially the production code for string conversion modules from Category 1 were staged and commited. Then the production code for the time conversion module from Category 2 was staged and commmited. Then the blackbox test fixtures for string conversions, and then time conversions were staged and commited. **All code was written on VScode**

Ethics and Professionalism

In the scenario that the time conversion modules are utilized in a large software application, such as the built in Google search engine conversions calculator. A lack of professionalism, in the case that an individual

or team do not implement proper test design and code for reasons such as poor time management; can lead to ethical concerns for the public. For example: A high school student using the time conversions calculator for a science project. A bug in the initial version of the calculator gave the wrong conversion output for minutes to seconds. Because of this the student ended up losing marks. Ultimately this is a minor opportunity loss, but still an ethical concern as result of lack of professionalism.

From the IEEE CS Code of Ethics, it is important that Software Engineers ensure that their *PRODUCT* meets the highest proessional standards. Meaning in this case, the code was to be professionally designed and tested meeting highest standards to avoid ethical issues.

In addition, Software Engineering managers must promote ethical approaches to *MANAGEMENT* of software development and maintenance. Meaning in this case, no matter how simple a piece of software was, it was important that Senior Management directed the thorough review and testing of the software.

Discussion

The program implements simple functionalities which perform various conversions: conversions of strings - uppercase/lowercase, and conversions of time - between units. These functionalities were modularized appropriately. In addition Version Control was also used to log project progress, and appropriate testing code was implemented to validate the correct behaviour of the modules. As required, the modules were designed and implemented to adhere to good modularity principles, and ethical concerns.

A major weakness of mine during this assignment was time management. My poor time management due to external factors hindered my project with reduced time to complete a formal planning process. This resulted in me rewriting my code twice due to poor planning, and in the end missing out on Whitebox Test Code implementation.

Overall this project allowed myself to further understand the intricate processes of Software Engineering project management. Giving me the opportunity to independently experience the process of planning, implementing, testing, and logging my project code. This project has also allowed me the opportunity to apply new concepts such as Modularity Principles, and consider Ethical Considerations in detail from a Software Engineering Standpoint.