FIT1008– Intro to Computer Science
Workshop Week 1

Semester 1, 2018

Objectives of this practical session

- For you to get to know each other and your demonstrator.
- To introduce you to some of the software you will be using in this unit.
- To introduce you to some of the requirements for documenting your code.

Pracs in FIT1008

Please read the "*FIT1008 PracGuide*" document on Moodle which provides more details of how pracs are organised for this unit [1].
As may not have had any lectures as yet, this prac does not require any preparation.
You may also benefit from watching the video guide to installing Python tools. The video describes how to get *PyCharm* installed on your own machine, which you should do before the first prac. [2]

[1] See here: https://moodle.vle.monash.edu/mod/resource/view.php?id=4698343

[2] See here https://moodle.vle.monash.edu/mod/url/view.php?id=4698329

Task 1

Using *PyCharm* code and run the following four Python modules. The four Python modules can also be downloaded from Moodle.

hello.py

```
1 name = input('Enter name (max 60 chars): ')
2 print('Hello ' + name + '. Welcome')
```

convert_temp.py

```
1 temp_C = int(input('Enter temperature in Celsius '))
2 temp_F = int(9*temp_C/5 + 32)
3 print('Temperature in Fahrenheit is ' + str(temp_F))
```

decimal_to_binary.py

```
def decimal_to_binary(decimal_number):
    my_list = []
    while decimal_number > 0:
        rem = decimal_number % 2
        my_list.insert(0, str(rem))
        decimal_number = decimal_number // 2
    return '0b' + ''.join(my_list)

number = int(input('Enter a number: '))
print('The binary representation is:')
print(decimal_to_binary(number))
```

decimal_to_hex.py

```
def decimal_to_hex(decimal_number):
    my_list = []
    while decimal_number > 0:
        rem = decimal_number % 16
        if rem > 9:
            if rem == 10:
                rem = 'A'
            elif rem == 11:
                rem = 'B'
            elif rem == 12:
                rem = 'C'
            elif rem == 13:
                rem = 'D'
            elif rem == 14:
                rem = 'E'
            else:
                rem = 'F'
        my_list.insert(0, str(rem))
        decimal_number = decimal_number // 16
    return '0x' + ''.join(my_list)
number = int(input('Enter a number: '))
print('The hexadecimal representation is:')
print(decimal_to_hex(number))
```

- Run these modules in interactive mode as well.
- Run the same code using a Jupyter Notebook.
- Discuss the differences between running the code in different ways.

Task 2

For each program you write in this unit you must provide documentation. Later in the unit, we will introduce testing, which has to be provided as well. The documentation must be written with the code, not in a separate file. Below you will find some examples, in terms of *docstrings*, of the type of descriptions you will need to provide for your code [3].
Here's one example with all the required items:

[3] See Guidelines here: `http://bit.ly/style_python`

```
def binary_search(the_list, item):
    """
    This function implements binary search
    :param the_list: a sorted list
    :param item: an item to search in the list
    :return: the index of item or -1 if item not in the list
    :raises: No exceptions
    :precondition: the_list must be sorted in increaseing order
    :complexity: best case O(1), worst case O(log n), where n is len(the_list)
    """
    lower = 0
    upper = len(the_list) -1
    while lower <= upper:
        mid = (lower + upper)//2
        if the_list[mid] == item:
            return mid
        elif the_list[mid] > item:
            upper = mid - 1
        else:
            lower = mid + 1
    return -1
```

For each of the Python modules in Task 1, add documentation using docstrings [4]. A good guide to pre and postconditions can be found on Moodle under week 0 [5].

[4] **Pycharm**: triple quote ⟦⏎⟧, then add complexity, exceptions and pre-conditions

[5] `https://moodle.vle.monash.edu/mod/resource/view.php?id=4698322`

Task 3

Later in this unit you will be implementing various data structures. A useful approach for testing your implementation is to use a menu, which allows you to test each of the functionalities of the data structure. Consider the following code which uses a menu to append items to a list, print a list out, sort a list, and to quit.

```python
def print_menu():
    print('\nMenu:')
    print('1. append')
    print('2. sort')
    print('3. print')
    print('4. quit')

my_list = []
quit = False
input_line = None

while not quit:
    print_menu()

    command = int(input("\nEnter command: "))

    if command == 1:
        item = input("Item? ")
        my_list.append(item)
    elif command == 2:
        my_list.sort()
    elif command == 3:
        print(my_list)
    elif command == 4:
        quit = True
```

Extend the above code so a user can perform the following commands on my_list using the menu:

- clear: which removes all the items in the list.
- reverse: which reverses the order of the items in the list.
- pop: which removes the last item of the list, and prints it.
- insert: which inserts an item before a given position in the list.

Task 4

This is the code review task.

Implement a program that provides a menu to convert numbers across different representations including decimal, binary and hexadecimal. Your menu should take a number in any format (of the ones mentioned above), and should be capable of converting it to any of the other formats (mention above). You can learn from the code given to convert decimals to binary and hexadecimal. Discuss with your class if necessary.