Swinburne University of Technology
Department of Computing Technologies

Software Testing and Reliability
SWE30009
Semester 2, 2024

Assignment 2

This is an individual assignment worth 20% of the total unit score.
Due date: 23:00 Monday 9 September 2024

DESCRIPTION

Program Under Test

Consider a program which splits an input list of integers into two lists of integers as output according to the following specifications:

- For the input list, the program is expected to accept a non-empty list of not more than 30 integers within the ranges of [-100, 100], where integer 0 is regarded to be negative in this assignment. The input list may contain duplicated integers (that is, the same integer may appear more than once in the input list).
- For the output lists, one of them contains only positive integers in ascending order without duplication of integers, and the other list contains only negative integers in ascending order without duplication of integers. Note that one of the output lists may be an empty list for some inputs.

For example, suppose that the input list is [10, -9, 15, 7, 10, -5, 28]. This list contains 7 elements which are all within the range of [-100, 100] with some integers like 10 appearing more than once in the list. This is a valid test case. Its relevant outputs are two lists, one of which is [-9, -5] (only negative integers, in ascending order without duplication), and the other list is [7, 10, 15, 28] (only positive integers, in ascending order without duplication. Since the input is restricted to contain no more than 30 integers whose values are within the range of [-100, 100], the computed outputs can be easily validated in a manual approach, and hence you can assume that this program has no test oracle problem.

A program-under-test is given in either Python or Ruby in the Appendix.

Assignment Requirement

In this assignment, you are required to do the followings:

A. TASKS

Task 1: You are required to ***construct six valid concrete test cases*** for this program, which serve for different testing purposes or objectives. Valid test cases are non-empty lists of integers with not more than 30 elements, where some elements could be duplicated and integer 0 is regarded as negative in this assignment.

Apart from explicitly describing your concrete test cases, you must ***describe and justify their purposes, and explicitly describe their relevant outputs***. For example, for the test case [13, 7, 9, 15]. Its outputs should be two lists, the [7, 9, 13, 15] and the empty list denoted by []. This test case serves to test the scenario that the output list of negative integers is empty.

Task 2: Suppose that due to testing resources, you cannot test the program with all your six test cases constructed above. You can only afford to ***test your program with one and only one test case***. What will be your choice of this single test case ***among the 6 test cases proposed by you in Task 1***? Explain and justify your choice.

Task 3: ***Conduct the testing with any one of two programs*** given in the Appendix with ***all your test cases in Task 1***. Note that the program may contain bugs. Report the individual test cases and the results, discuss the insights of test outcome including the suggestion to improve the program under test from the test finding.

B. REQUIREMENTS

- **You need to prepare a self-contained and complete report for addressing these three tasks**. Your report must use single column, 12-point font size on A4 papers. Your report must contain ***no more than three pages***. Coverage page is not required, but your submission must contain ***your full name and your student number*** on the first line of its first page.

- **You must submit the Assignment 2 report as a single PDF file** (which must have the filename specified in this format as "Assignment2-YourStudentID-YourFullName.pdf", for example, Assignment2-12345678-JohnNash) before the due date using the Submission System for Assignment 2 in Canvas available to you.

C. INSTRUCTIONS

In "Lecture 3.pdf", we discuss how to use the information about the inputs and outputs for a program to define test cases without referencing to the codes of the program under test. You are required to use this approach to complete the first two tasks. For Task 3, you must focus on testing the assignment2.py or assignment2.rb.

MARKING CRITERIA

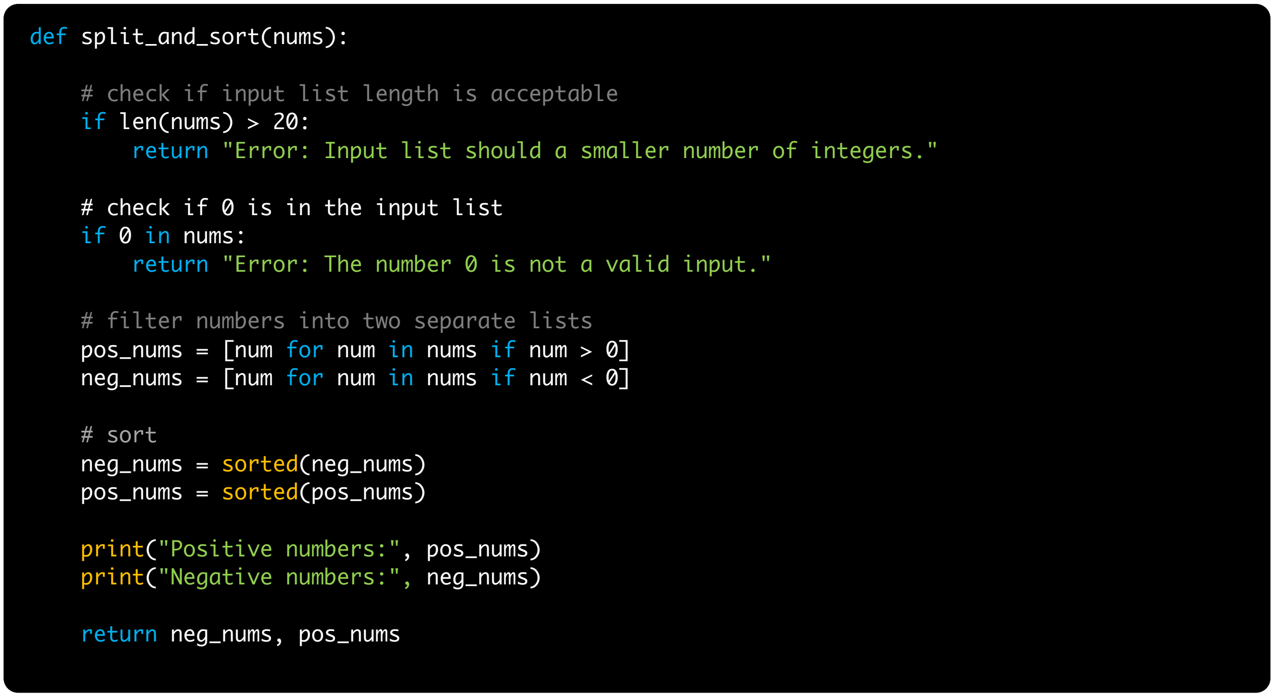
- **A maximum of 40 marks** for Task 1 which include 5 marks for each test case description, its output, its justification and 10 marks for the quality and comprehensiveness of its description and presentation.
- A **maximum of 20 marks** for the explanation and justification of your choice for Task 2.
- **A maximum of 30 marks** for the test case description, the test execution and results, the insights of discussion as well the program improvement for Task 3.
- **A maximum of 10 marks** for the presentation, structure, comprehensibility, completeness and cohesion of report, as well as the compliance with the submission requirements.
- **Penalty** will be applied for late submission and plagiarism. Refer to the Unit Outline for the policy on late submission and plagiarism.
- You are **not allowed to use of any artificial intelligence tool** such as ChatGPT, Bard to complete any part of your assignment submission (including report contents, code, and test script). It will be treated as plagiarism and will be penalized.

APPENDIX

You can use either Python or Ruby program below for your Task 3 for testing. In other words, you can use either option A or option B for your work. They can be downloaded in Canvas, too.

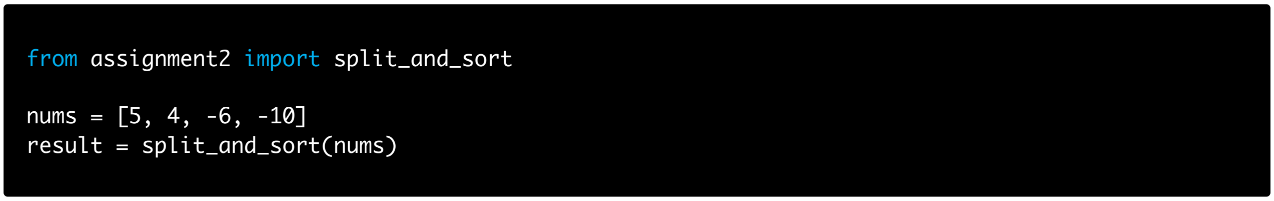
Option A: Python program

Program under test in Python **assignment2.py**



```python
def split_and_sort(nums):

    # check if input list length is acceptable
    if len(nums) > 20:
        return "Error: Input list should a smaller number of integers."

    # check if 0 is in the input list
    if 0 in nums:
        return "Error: The number 0 is not a valid input."

    # filter numbers into two separate lists
    pos_nums = [num for num in nums if num > 0]
    neg_nums = [num for num in nums if num < 0]

    # sort
    neg_nums = sorted(neg_nums)
    pos_nums = sorted(pos_nums)

    print("Positive numbers:", pos_nums)
    print("Negative numbers:", neg_nums)

    return neg_nums, pos_nums
```

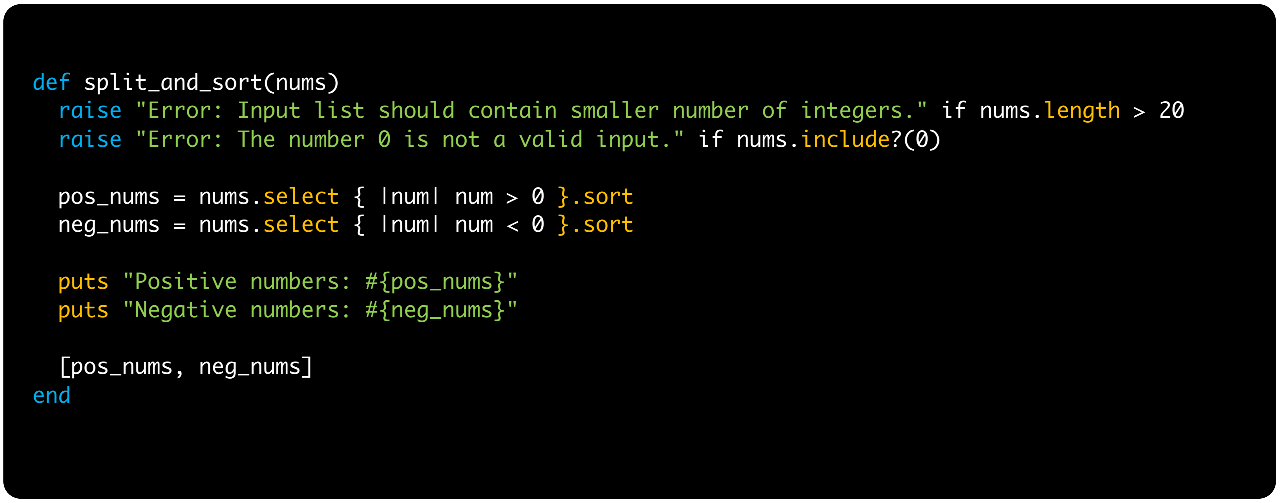
To test this program, we can use a Python script **test.py.**



```python
from assignment2 import split_and_sort

nums = [5, 4, -6, -10]
result = split_and_sort(nums)
```

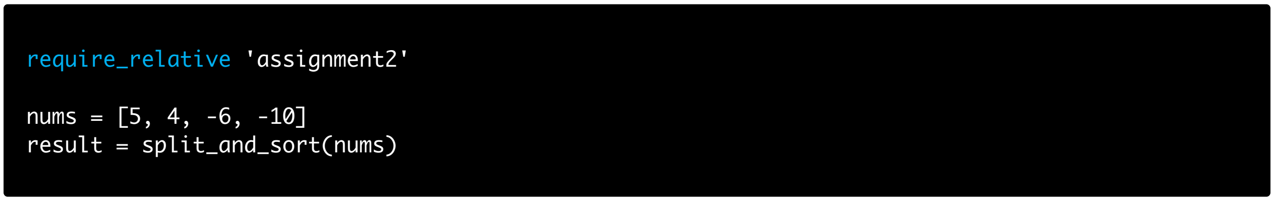
Option B: Ruby program

Program under test in Ruby **assignment2.rb**.



```ruby
def split_and_sort(nums)
  raise "Error: Input list should contain smaller number of integers." if nums.length > 20
  raise "Error: The number 0 is not a valid input." if nums.include?(0)

  pos_nums = nums.select { |num| num > 0 }.sort
  neg_nums = nums.select { |num| num < 0 }.sort

  puts "Positive numbers: #{pos_nums}"
  puts "Negative numbers: #{neg_nums}"

  [pos_nums, neg_nums]
end
```

To test this program, we can use a Ruby script **test.rb.**



```ruby
require_relative 'assignment2'

nums = [5, 4, -6, -10]
result = split_and_sort(nums)
```