Assignment 1: Basics and Testing

Formative, Weight(10%), Learning objectives (1,2,3),
Abstraction (4), Design (4), Communication (4), Data (5), Programming (5)

Due date: 11:59pm, 17 August 2018, Weight: 10.0 % of the course

1 Overview

Assignments should be done in groups consisting of four (4) students. If you have problems finding a group partner use the forum to search for group partners or contact the lecturer.

Reminder: unless we explicitly allow it for specific exercises, you must not copy and paste from websites, scientific articles, your friends, ...

For this assignment you will have to hand in a report with the answers to the questions, and describing what you have done. In case you decide to write your own code, you will need to include the code in your submission.

2 Assignment

Exercise 1 *Team work: who has done what? (zero points)*

We'd like each team member to write one paragraph about what he or she has contributed to this assignment. We will not mark this, and it will not have any effect on the marking of the other exercises. You might now ask "why do this then?" — well, through this no-stakes approach, we'd like to encourage self-regulation within the group and cooperative learning. You can't lose, you can only win.

Exercise 2 *SBSE problems, our choice (8 points)*

Software development effort estimation is the process of predicting the most realistic amount of effort (expressed in terms of person-hours or money or software size) required to develop or maintain software based on incomplete, uncertain and noisy input.

The problem:
You are given a set of n well-formed equations (i.e., E_1, E_2, E_3, ..., E_n) that can be used to produce cost predictions and a set of real-world data. From these, you need to select the equation E_i with the best predictive capability.

Answer the following questions:

1. What are the objectives?

2. How can an example solution look like, i.e., what is the problem representation?
3. How would you measure the fitness of a solution for a single objective?
4. Which search algorithm would you choose? Why?

In your report, be comprehensive in your answers.

Exercise 3 *SBSE problems, your choices (32 points)*

Describe four (4) other software engineering problems (different from the one in Exercise 1) to which SBSE techniques can be applied to.

In all four cases:

1. describe the problem,
2. describe why it is fit for SBSE (hint: think about the search space),
3. describe a possible problem representation (including how you would create neighbouring solutions), and
4. propose a fitness function.

Notes:

- In your report, be comprehensive in your answers. Think of a real-world situation where Mahmoud (who will mark your assignment) is your team leader who needs to be convinced.
- The problems can be (but don't have to be) from the papers that you will be presenting and discussing in the spotlight talks and essays. This is to encourage you to start early with your spotlight talks and essays :) If you decide to go with this option, then please provide the URL to the paper.
- Remember that fundamental combinatorial problems, such as travel salesman and knapsack problems, are not SBSE problems.

Exercise 4 *Artificial interaction testing (20 points)*

Part 1

Look up "combinatorial interaction testing". Provide a definition and a brief (but complete) example.

You can use online resources, such as Wikipedia, without losing points. However, it is mandatory that you properly reference the resource. For more information on proper referencing, see http://libguides.adelaide.edu.au/referencing.

Part 2

Given is a Car Model, where car configurations can be customised by selecting components from options. Generate a comprehensive 4-way interaction test suite, which consists of as many valid tests as possible, for the following set of configurations for the following Car Model:

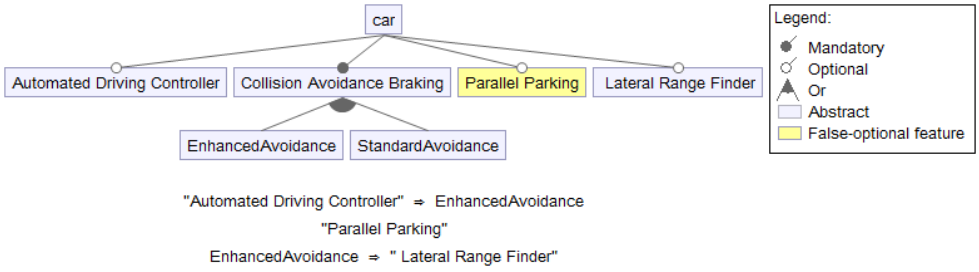
- Automated Driving Controller: included, none
- Collision Avoidance Braking: StandardAvoidance, EnhancedAvoidance
- Parallel Parking: included, none
- Lateral Range Finder: included, none

It is important that your test suite respects our following constraints:

- Parallel parking must always be included.
- If automated driving controller is included, then collision avoidance braking must be enhanced.
- If collision avoidance braking is enhanced, then lateral range finder must be included.

Inn your report, include a description of how you generated the tests and how you considered the constraints. This description should be no longer than two pages.

Note: You can see this as a scenario from Software Product Line Engineering (SPLE), and your task is to find the valid combinations. In SPLE, the goal is to create collections of similar products from a shared set of assets. Here is a visualisation of the overall situation, created in Eclipse:



False-optional feature means that it is declared optional, but it is always selected if its parent (car) is selected.

Exercise 5 *Real-world interaction testing (40 points)*

Part 1

Look up "pairwise testing". Provide a definition and a brief (but complete) example.

Your example needs to be different from the one you are submitting as an answer for Exercise 3.

You can use online resources as stated in Exercise 3, if you following our referencing requirement.

As before, be comprehensive in your answers. Think of a real-world situation where Mahmoud (who will mark your assignment) is your team leader who needs to be convinced.

Part 2

Django is a very popular Python-based framework for rapid development of web-based applications. Among popular sites using it are: Instagram, Mozilla and The Washington Times. Django is written in Python and comes with a global settings file that contains parameters that can be configured in any Django-based web application.

Given the settings file at https://github.com/django/django/blob/master/django/conf/global_settings.py, answer the following questions:

1. How many parameters are there in the Django settings file?
2. List all the Boolean parameters.
3. Create a pairwise test suite for all Boolean parameters, and describe how you have done this. Aim at creating a minimal pairwise test suite.

Notes:

- You are not allowed to use any existing software for the generation of the test suite.
- Provide the list of tests as an additional plain text file (no PDF/DOCX/...). In this file, list one test case per line, in any human-readable encoding you like.
- It is NOT necessary to provide executable code that runs Django.

3 General procedure for handing in the assignment

Work should be handed in using the course website. The submission should include:

- pdf file of your solutions for theoretical assignments
- all source files (if you created any)
- descriptions as required in the statement of the exercises
- a file name README.txt that contains instructions to run the code (if any), the names, student numbers, and email addresses of the group members

Note: as there is little to no coding involved in this assignment, there will be NO progress presentation session for this assignment.