

CSC 7003 : Basics of Software Engineering

J Paul Gibson, A207

`paul.gibson@int-edu.eu`

<http://www-public.it-sudparis.eu/~gibson/Teaching/CSC7003/>

Team Project

<http://www-public.it-sudparis.eu/~gibson/Teaching/CSC7003/TeamProject.pdf>

The 15-puzzle: planning a software project

This assignment is about planning a software development project, organising teams and scheduling tasks

You will be judged on your:

- Planning (40%)
- Implementation of the plan (40%)
- Evaluation/Evolution of the plan (20%)

It is your job to:

- Allocate roles
- Choose a development process life cycle
- Organise tasks into a schedule coherent with roles/life cycle

The 15-puzzle: planning a software project

The problem is to implement a 15-puzzle system.

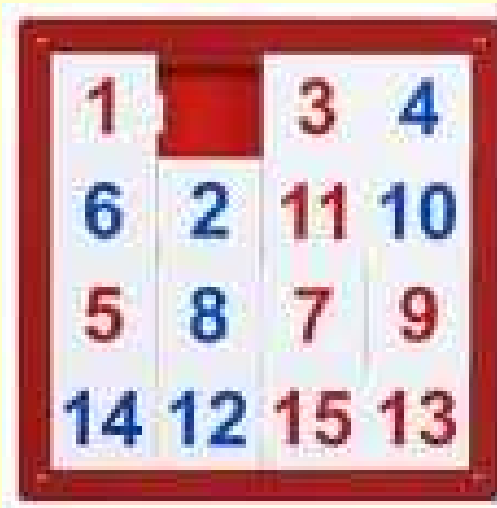
**Final
Position**



**Impossible
Position**



**Solvable (valid)
Position**



The 15-puzzle: planning a software project

The problem is to implement a 15-puzzle analyser.

Your system is to be parameterised by a *fitness function*:

Input – board state

Output – value between 0 and 1

- 0 is to represent the board in a solved position
- 1 to represent the board in its ‘most mixed up’ position
- The ordering of the fitness function must rank board positions according to their ‘closeness to being solved’

You must code a generic puzzle solver that can find a solution (sequence of moves) that returns the puzzle to the solved state from any given valid input state.

The 15-puzzle: planning a software project

Your generic solution must follow the following steps:

Repeat

- For any given position look for the shortest sequence of moves that improves the value of the fitness function (towards the solution)
- Carry out the moves

Until Solved

Simulate the solution found (sequence of moves) for validation by a human

Test the solution automatically

Output the length of the sequence, and the length of the longest subsequence that was taken during a single step of the algorithm loop.

The 15-puzzle: planning a software project

Analysis and tests:

You must analyse 3 (significantly) different fitness functions with respect to the (sub)sequence solution length values:

- Which fitness functions find the shortest solution?
- Which fitness functions find a solution without having to look too deep during each loop step?
- What is the complexity of each of the 3 solutions (time/memory)?