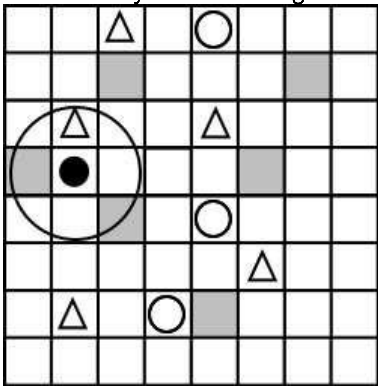
Practical 9 (due 2022-05-13 @ 09:00)

The purpose of this assignment is to introduce the type of problems presented during the semester test 2 and exam. 2021 ST2 PAPER MORNING.

Please note: Submissions will be checked for originality. If you use someone else's code or code is taken from the Internet, then your prac will come under scrutiny for a potential copy, which may result in zero marks being awarded along with potential further disciplinary action.

Recipe for Success

The Utopian Softworx Corporation is hoping to cash in on the success of the survival crafting genre of games. You will need to create a text base prototype of such a game for upper management. While the final version will be multi-player your prototype needs to only be single player. In the game the player moves around a two dimensional environment in a turn based manner collecting materials. Some materials may be collected by simply moving over them while others require specialised tools to be created first. The goal of the game is to gather the tools required to make a fire before the day ends and night falls.



Player (solid black circle), empty space (white squares), bushes (dark squares), trees (triangles), flint (white circles), collection radius (large circle).

In the game you will need to move a player controlled character around a two-dimensional playing area. Your logic must be placed in the `RecipeSpace` namespace.

Initialisation:

- The size of the environment, chance of bushes, number of trees, number of flint stones, and number of turns until nightfall are specified via command line arguments.
- The world is filled in the following order. Different entities cannot occupy the same position in the world.
  - The player starts in the position closest to the centre of the game world ◦ A user-specified fixed number of the following are placed in the world:
    - Trees
    - Flint
  - The following are placed with a user-specified fixed probability per squa
    - Bushes
  - The player starts out carrying 0 units of the following items: sticks, flint, logs, axes, and fire kits.

Movement:

- The player may move one step in each direction (including diagonals). The player may not move outside of the game area.
- If the player moves onto a square containing a bush the bush is removed from the world and the player gains one unit of sticks
- If the player moves into a square containing a tree nothing happens.

- If the player moves into a square containing flint they pick the flint up and it is removed from the world.
- Instead of moving the player may choose to craft (create) an item if they are carrying the right amount of materials. These materials are consumed and the crafted item is added to the player's collection.
    - 1 stick and 1 flint creates 1 axe
    - 2 sticks, 1 flint, and 3 logs creates a fire kit
- If the player is carrying an axe they may choose to chop wood. This removes all the trees in a one square radius around the player and adds one log to the player's collection for each tree removed.
- If the player is carrying a fire kit they may choose to light a fire which appears in the game world next to them.

Update:

- After each turn the time until nightfall decreases by 1

End-game:

- The game ends in failure when night falls
- The game ends in victory if the player lights a fire

Any submission that does not compile will be capped to 40%.

Consider the competencies as laid out in the mark sheet.

- C0 – Create a program design. The entire design including UML must model the initialisation function.
- C1 – Use your knowledge of basic C++ program structure and make sure to utilise the appropriate system libraries.
- C2 – Your program must be readable by human beings in addition to compiler software.
- C3 – Demonstrate your knowledge of the divide and conquer principle using functions.
- C4 – Your program must make use of programmer defined source code libraries.
- C5 – Create a menu system which will ask the user which action they wish to take.
- C6 – The user must provide the required inputs used by the game (with error handling).
- C7 – Provide assertion based error handling as well as conventional error handling.
- C8 – Random numbers are used when initialising the 2D array
- C9 – Use dynamic 2D arrays to implement your simulation. The game state must be output to screen using printable ASCII characters.
- C10 – Pay careful attention the handling of the movement, update rules, and end conditions for the game.

UNIVERSITY OF JOHANNESBURG

Original Marksheet

Mark sheet		
Competency	*Description*	*Result*
C0	Program Design	/10
C1	Boiler plate code • Standard namespace (1) • System library inclusion (3) • Indication of successful termination of program (1)	/5
C2	Coding style • Naming of variables (1) • Indentation (1) • Use of comments (1) • Use of named constants (1) • Program compiles without issuing warnings (1)	/5
C3	Functional Abstraction • Task decomposition (5) • Reduction of repetitive code (5)	/10
C4	Separate Compilation • Header file (1) • Guard conditions (2) • Inclusion of header file (1) • Appropriate content in header file (1) • Use of programmer defined namespace (5)	/10
C5	User Interaction • Menu System (5) • Appropriate use of input, output and error streams (5)	/10
C6	Command Line Argument Handling: • Appropriately overloaded main function (1) • Handling incorrect argument counts (1) • Use of supplied arguments (3)	/5
C7	Error Handling • Use of assertions (2) • Use of conventional error handling techniques (3)	/5
C8	Pseudo-random number generation (5)	/5
C9	Dynamically allocated two-dimensional array handling • Structures (5) • Allocation (5) • Initialisation (5) • Deallocation (5)	/20
C10	Algorithm implementation • Logical Correctness (5) • Effectiveness / Efficiency of approach (5) • Correct output (5)	/15
B	Bonus	/10
Total:		**/100**