Life (Part 1) - CRA & Statement of Completeness

CAB201 - Programming Principles

Last Name: <Madigan>
First Name: <Johnny>
Student ID: <nxxx>

Score: 100.00
Grade: 35.00%

This document will need to accurately state the elements of your submission that have been completed. Enter numbers into each of the criteria that reflect how well you think you did. Include any limitations, bugs, logical/run-time error or general comments in the comment section at the end of each section. You should use this as a 'checklist' for completing your assignment.

Submission Items Each of the items below are required for submission, failure to submit any of these items will result in a point deduction. The amount deducted will be proportional to the amount of incomplete work.	**Score**	**Max**
Progress Submission	5	**5**
CRA & Statement of Completeness	1	**1**
User Manual (`README.md`)	2	**2**
Total	**8**	**8**
Comments		

Command Line Arguments To gain points for this section, your program must be able to correctly interpret command line arguments. Generally, each command line argument will be tested for its flag name, default value and validation.		**Score**	**Max**
General	Multiple flags can be used at once	3	**3**
	Missing number of paramters (for any given option) are reported	3	**3**
Default Values	**Step Mode**: OFF	1	**1**
	Periodic Behaviour: OFF	1	**1**
	Random Factor: 0.5 (50%)	1	**1**
	Update Rate: 5 updates/second	1	**1**
	Generations: 50	1	**1**
	Input File: N/A	1	**1**
	Dimensions: 16 x 16	1	**1**
Usage and Effect	**Step Mode**: `-step` correctly enables step mode	2	**2**
	Periodic Behaviour: `-periodic` correctly enables periodic behaviour	2	**2**
	Random Factor: `-random` correctly changes the random factor	2	**2**
	Update Rate: `-max-update` correctly changes the update rate	2	**2**
	Generations: `-generations` correctly changes the number of generations	2	**2**
	Input File: `-seed` correctly sets the seed file path	2	**2**
	Dimensions: `-dimensions` correctly changes the rows and columns	2	**2**
Validation	**Random Factor**: Floating point values between 0 and 1 (inclusive)	1	**1**
	Update Rate: Floating point values between 1 and 30 (inclusive)	1	**1**
	Generations: Integer values above 0	1	**1**
	Input File: Valid paths with a `.seed` file extension	1	**1**
	Dimensions: Integer values between 4 and 48 (inclusive)	1	**1**
Total		**32**	**32**
Comments			

Functionality and Flow	

To gain points for this section, your program must behave in correctly and in accordance to the specification.		Score	Max
Seed File	Seed file can be read by program, and override the random seed functionality	2	**2**
	Seed file cells are presented in the correct orientation	2	**2**
Flow	Program starts by reporting the success or failure of interpretting the command line arguments	3	**3**
	Program displays the runtime settings before beginning the simulation	3	**3**
	The spacebar key is exclusively used to progress the program	3	**3**
Simulation	The generation number is displayed to the bottom right of the grid.	2	**2**
	When the simulation is complete, the grid will display as complete and wait for the user before clearing the screen	2	**2**
	The simulation behaves acording to the rules of *Life*	10	**10**
	The simulation can be controlled by generation in step mode	2	**2**
	The simulation can exhibit periodic behaviour	5	**5**
Presentation	Warning/error messages are descriptive and insightful	3	**3**
	Console output is clearly presented and uncluttered	3	**3**
Total		**40**	**40**
Comments			

Code Quality To gain points for this section, you must maintain good code quality throughout your whole project. You should follow the guidelines and conventions presented in the CAB201 C# Programming Style Guide to help you meet these criteria, **Important:** Your target reader is a programmer, not an absolute beginner.	Score	Max
Naming: Maintained, consistent and clear standard in variable, method and class naming.	3	**3**
Magic Numbers: Magic numbers have been replaced with appropriately named constants.	2	**2**
Variable Declarations: Varaibles are declared with minimum scope and global declarations are avoided where appropriate.	2	**2**
Format: Consistent and appropriate white spacing, line length, indentation, bracing, and separation into files within the project.	3	**3**
Comments: All classes and methods are appropriately commented. Use of in-line comments to explain complex (non-obvious) code. In-line comments are not excessive.	4	**4**
Methods: Methods are single purpose and clear, and code is reasonably efficient and succinct.	3	**3**
Code reuse: No unnecessarily long or repeated code. No redundant methods.	3	**3**
Total	**20**	**20**
Comments		

Life (Part 2) - CRA & Statement of Completeness

CAB201 - Programming Principles

Last Name: <Madigan>
First Name: <Johnny>
Student ID: <nxxx>

Score: 100.00
Grade: 35.00%

This document will need to accurately state the elements of your submission that have been completed. Enter numbers into each of the criteria that reflect how well you think you did. Include any limitations, bugs, logical/run-time error or general comments in the comment section at the end of each section. You should use this as a 'checklist' for completing your assignment.

Submission Items Each of the items below are required for submission, failure to submit any of these items will result in a point deduction. The amount deducted will be proportional to the amount of incomplete work. For each of the report sections you must discuss at least one good example of how you have used each of the concepts in your code and how it has improved your code quality to receive full marks.		**Score**	**Max**
Progress Submission		5	**5**
CRA & Statement of Completeness		1	**1**
User Manual (`README.md`)		2	**2**
Project Report	Encapsulation	3	**3**
	Inheritance	3	**3**
	Polymorphism	3	**3**
	Exception Handling	3	**3**
Total		**20**	**20**
Comments			
README has been updated for the new features and also a lot cleaner.			

Command Line Arguments To gain points for this section, your program must be able to correctly interpret command line arguments. Generally, each command line argument will be tested for its flag name, default value and validation. It is assumed that the command line arguments introduced in Part 1 are completely functional (points will only be awarded for arguments introduced in Part 2).			**Score**	**Max**
General	Multiple flags can be used at once		2	**2**
	Missing number of paramters (for any given option) are reported		2	**2**
Default Values	**Neighbourhood**: 1st order Moore (centre excluded)		1	**1**
	Survival Rule: 2 or 3 neighbours		1	**1**
	Birth Rule: 3 neighbours		1	**1**
	Memory: 16 generations		1	**1**
	Output File: N/A		1	**1**
	Ghost Mode: OFF		1	**1**
Usage and Effect	**Neighbourhood**: --neighbour correct sets the type order and centre counting for the neighbourhood		2	**2**
	Survival Rule: --survival correctly sets the number of neighbours required for survival		2	**2**
	Birth Rule: --birth correctly sets the number of neighbours required for birth		2	**2**
	Memory: --memory correctly sets the number of stored generations		2	**2**
	Output File: --output correctly sets the output file path		2	**2**
	Ghost Mode: --ghost correctly enables ghost mode		2	**2**
	Neighbourhood	Type is one of "moore" or "vonNeumann" (ignore case)	1	**1**
		Order is between 1 and 10 (inclusive) and less than half of the smallest dimension (rows or columns).	1	**1**

Validation		Centre-counting is one of "true" or "false" (ignore case)	1	**1**
	Rules	Any number of positive integers or positive integers separated by ellipses	1	**1**
		Supplied values are less than the number of neighbouring cells.	1	**1**
	Memory: Integer values between 4 and 512 (inclusive)		1	**1**
	Output File: Valid paths with a `.seed` file extension		1	**1**
Total			**29**	**29**
Comments				
Addressing the feedback about the .seed file, issue was due to program only being able to read Mac file paths (could not differentiate between forward slashes & back slashes. After some testing, fixed the issue by adding a more universal approach to reading files.				

Functionality and Flow To gain points for this section, your program must behave in correctly and in accordance to the specification. Note that criteria carried over from Part 1 have had their weighting reduced so that you are marked on the new elements.		**Score**	**Max**
File I/O	Seed files formatted with #version=2.0 are read correctly.	2	**2**
	Cells that would exist outside of the bounds of the universe are handled elegantly (the program doesn't crash and issues are reported).	2	**2**
	Data is written to the specified output file at the end of the simulation	2	**2**
	The output data is correctly formatted in accordance to the #version=2.0 formatting.	2	**2**
Flow	Program starts by reporting the success or failure of interpretting the command line arguments	3	**3**
	Program displays the runtime settings before beginning the simulation	2	**2**
	The spacebar key is exclusively used to progress the program. Holding down the spacebar should not progress the program.	2	**2**
Simulation	The game behaves correctly with non-standard neighbourhoods	5	**5**
	The game behaves correctly with non-standard birth/survival rules	5	**5**
	The simulation can render the the game using ghost mode correctly.	5	**5**
Presentation	Warning/error messages are descriptive and insightful	3	**3**
	Console output is clearly presented and uncluttered	3	**3**
Total		**36**	**36**
Comments			

Code Quality To gain points for this section, you must maintain good code quality throughout your whole project. You should follow the guidelines and conventions presented in the CAB201 C# Programming Style Guide to help you meet these criteria, **Important:** Your target reader is a programmer, not an absolute beginner.	**Score**	**Max**
Naming: Maintained, consistent and clear standard in variable, method and class naming.	2	**2**
Magic Numbers: Magic numbers have been replaced with appropriately named constants.	1	**1**
Variable Declarations: Varaibles are declared with minimum scope and global declarations are avoided	2	**2**
Format: Consistent and appropriate white spacing, line length, indentation, bracing, and separation	2	**2**
Comments: All classes and methods are appropriately commented. Use of in-line comments to explain complex (non-obvious) code. In-line comments are not excessive.	3	**3**
Methods: Methods are single purpose and clear, and code is reasonably efficient and succinct.	2	**2**
Code reuse: No unnecessarily long or repeated code. No redundant methods.	3	**3**
Total	**15**	**15**
Comments		