**Visualizer**

For your convenience we created a visualizer, which you can use to simulate the result from your solution.

**Running the visualizer**

The following items are required to run the visualizer:

* archive with the project code, uploaded to the codeit.bg site (link: <http://codeit.bg/bul/rounds/listTasks/39>);
* Microsoft Visual Studio (version 2012 recommended). You can download the Express version for free from this link: <http://www.microsoft.com/en-gb/download/details.aspx?id=34673>.

After unpacking the archive in a directory of your choice, open the file “**s4\_visualizer\s4\_visualizer.sln**”. Wait until Microsoft Visual Studio finishes loading and run the visualizer by pressing “**ctrl + f5**”.

The visualizer’s working directory is **s4\_visualizer\bin\Debug**.

**Instructions**

 In the field **Solution** you should type the full path to the executable file for your solution. During the simulation this file will be copied to **s4\_visualizer\bin\Debug\wkdir\\_sol.exec**.

 The field **Command** contains the command used to run your solution. By default it is "**\_sol.exec"** (i.e. it just runs your executable).

From the drop-down menu of the field **Tests** you can choose the type of the input file:

* **file** – your solution will be tested with a file of your choice. You must type in the full path to the input file in the field **<path>**. During the simulation this file will be copied to **s4\_visualizer\bin\Debug\wkdir\flow.in.**
* **generate** – allows you to generate an input file. In the field **<seed>** you can type the seed to be used for the random number generator, or you can leave it empty to have a string be generated for you. In the fields **Grid Size**, **Flows**, **Blocked** you should type in the size of the dessert areas, the number of flows and the number of cells containing bases. During simulation, an input file **s4\_visualizer\bin\Debug\wkdir\flow.in** will be generated.

The button **Run** starts the simulation, provided you have chosen a correct executable and input file. This is the general lifecycle of the simulation (assuming no errors appear):

* the input file is copied and is checked for correctness using **input-checker.exe** or an input file is generated;
* the chosen executable file is copied and run in **s4\_visualizer\bin\Debug\wkdir;**
* the resulting output file is checked using **output-checker.exe;**
* visualization starts.

Visualization starts in a paused state. Using the buttons on the last row you can issue the following commands:

* **PA** -pause/resume – Pauses or resumes the simulation. While the simulation is running, periodically a command is read from the output file and the corresponding cell from the flow is changed. You can change the speed with which operations are read from the **steps/sec** field, by either choosing a new speed from the drop-down menu or typing a number in the field (max 1000), and pressing the **Set** button. The grid is visualized once every 0.25 seconds, no matter how many operations per second are done.
* **NE** – next – (Only if the simulation is in paused state). Simulates a number of operations from the output file, depending on the arguments given in the neighboring filed:
	+ **<empty>** - do one operation;
	+ **<integer n>** - do **n** operations;
	+ **f** – do all operations for the current flow.
	+ **e** – do all operations.
* **ST –** stop – stops the current simulation.

In addition, while the simulation is paused, you can click on a cell to get more detailed information about it – its coordinates (row, column) and the flow it is part of.

Along with the visualizer we provide you a few utility programs you could you, located in the folder **s4\_visualizer\bin\Debug\wkdir:**

* **generator.exe** (usage: generator.exe <seed> <gridSize> <nFlows> <nBlocked>) – generates an input file **flow.in**;
* **input-checker.exe** (no args)–checks weather the file **flow.in** is a valid input file. If it is, the program returns with return code 0. If it is not, the program returns with return code 1 and a message is printed to stdout;
* **output-checker.exe** (no args) – works the same way as input-checker.exe, except if checks weather **flow.out** is a valid output file.