Langton's Ant Simulator Report

What is Langton's Ant?

The Langton ant is a simple cellular automaton invented and described by Chris Langton in 1986. It is based on a two-dimensional board in which each cell takes on white or black. In each step, one cell called an "ant" is distinguished, which, in addition to its color, also has a specific direction in which it moves.

Initial setup

The board is divided into square cells.

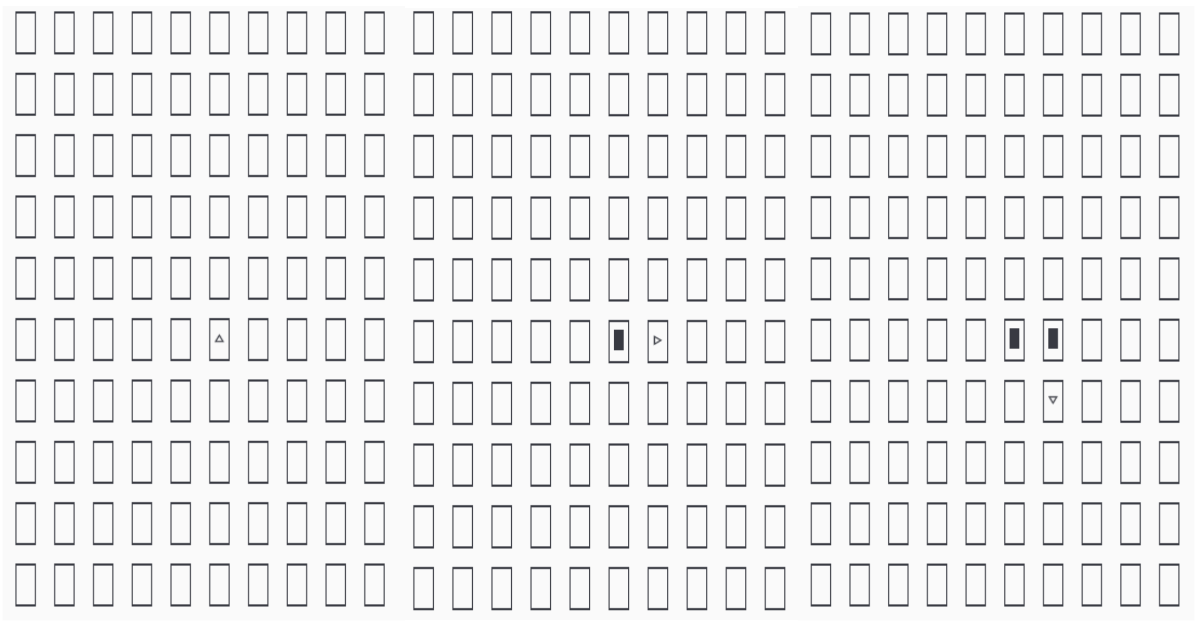
Each cell can take one of two colors: white or black.

The Langton ant takes off on one of the cells. Let's assume that it is a cell located as close to the center of the board as possible.

Movement rules

The ant walks on a two-dimensional grid. It can move in one of 4 directions (up, down, left, right), according to the following rules:

- The ant is in the white cell, it makes: a 90-degree turn to the right, changes the color of the cell to the opposite, moves one cell forward
- The ant is in the black cell, it makes: a 90-degree turn to the left, changes the color of the cell to the opposite, moves one cell forward

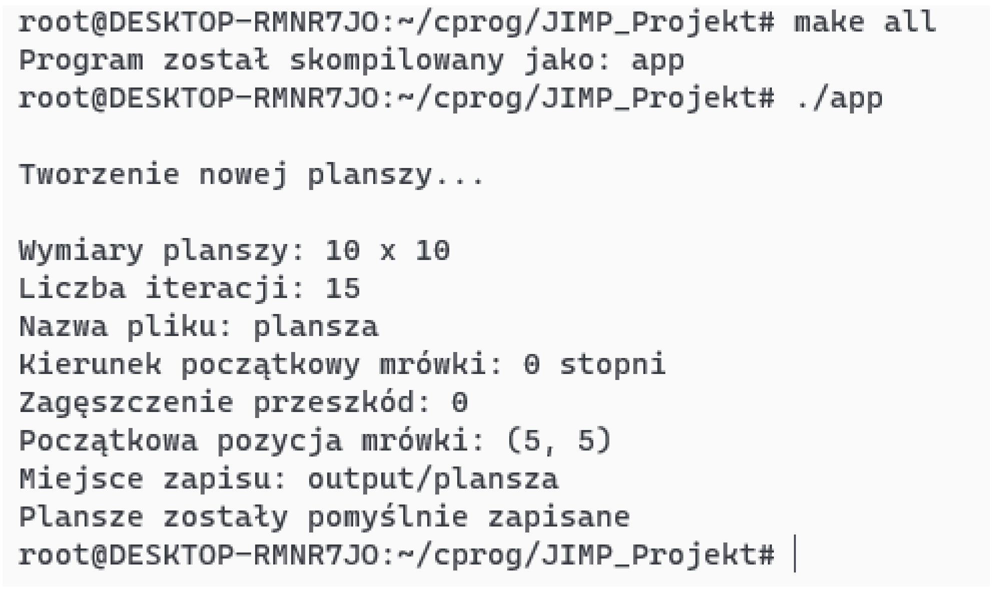


Figs 1-3. Example simulator call for two iterations

Invoking the program

The program is compiled with the command: **make all**

Then, to run the program, type: **./app**

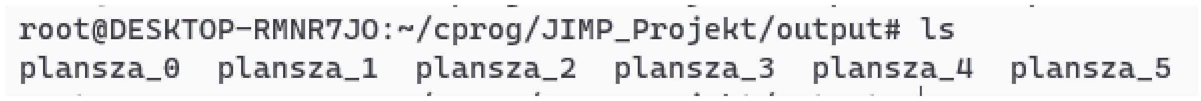


```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# make all
Program został skompilowany jako: app
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# ./app

Tworzenie nowej planszy...

Wymiary planszy: 10 x 10
Liczba iteracji: 15
Nazwa pliku: plansza
Kierunek początkowy mrówki: 0 stopni
Zagęszczenie przeszkód: 0
Początkowa pozycja mrówki: (5, 5)
Miejsce zapisu: output/plansza
Plansze zostały pomyślnie zapisane
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt#
```

Fig 4. Compiling and invoking the program

If the program was called without arguments, the board was created according to the default settings (see above above).

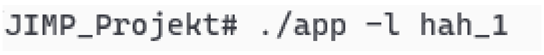
Boards are saved in the output folder by default.



```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt/output# ls
plansza_0  plansza_1  plansza_2  plansza_3  plansza_4  plansza_5
```

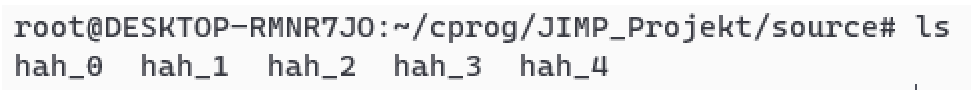
Fig. 5. The output folder containing the created boards

To attach your own board to the program, use the -l <plansza_indeks> command, and place the board in the source directory.



```
JIMP_Projekt# ./app -l hah_1
```

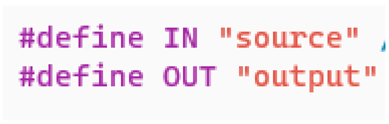
Fig 6. Calling a program with its own board.



```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt/source# ls
hah_0  hah_1  hah_2  hah_3  hah_4
```

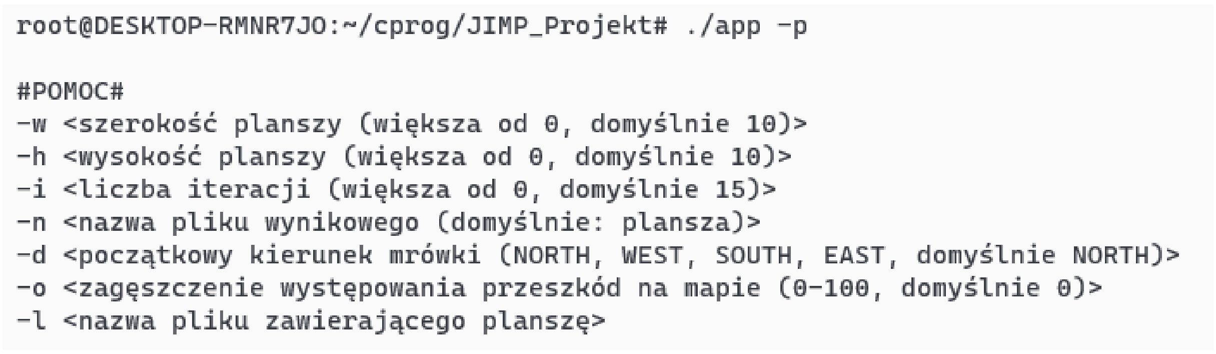
Fig 7. Source folder, containing the board ready to load

You can change the storage or reading location by editing the 6th and 7th lines in the main.c file, respectively.



```
#define IN "source"
#define OUT "output"
```

Information about the available arguments of the call is obtained by typing the argument -p



```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# ./app -p

#POMOC#
-w <szerokość planszy (większa od 0, domyślnie 10)>
-h <wysokość planszy (większa od 0, domyślnie 10)>
-i <liczba iteracji (większa od 0, domyślnie 15)>
-n <nazwa pliku wynikowego (domyślnie: plansza)>
-d <początkowy kierunek mrówki (NORTH, WEST, SOUTH, EAST, domyślnie NORTH)>
-o <zagęszczenie występowania przeszkód na mapie (0-100, domyślnie 0)>
-l <nazwa pliku zawierającego planszę>
```

Fig 8. Menu Help

Module division of the program

The program has been divided into 4 modules, i.e.: board, logic, arguments and the last main module, which calls each of the other modules.

- MODULE **board**
  It contains the structure of the entire board (the size of the board, the position of the ant, the direction in which the ant moves, the structure informing what color a given cell on the board is)
  It consists of the following functions:
  - createBoard - initializes the structure of the board
  - printBoard – prints the board to the output
  - saveBoardToFile – saves the board to a file
  - loadBoardFromFile – reads the board from the file

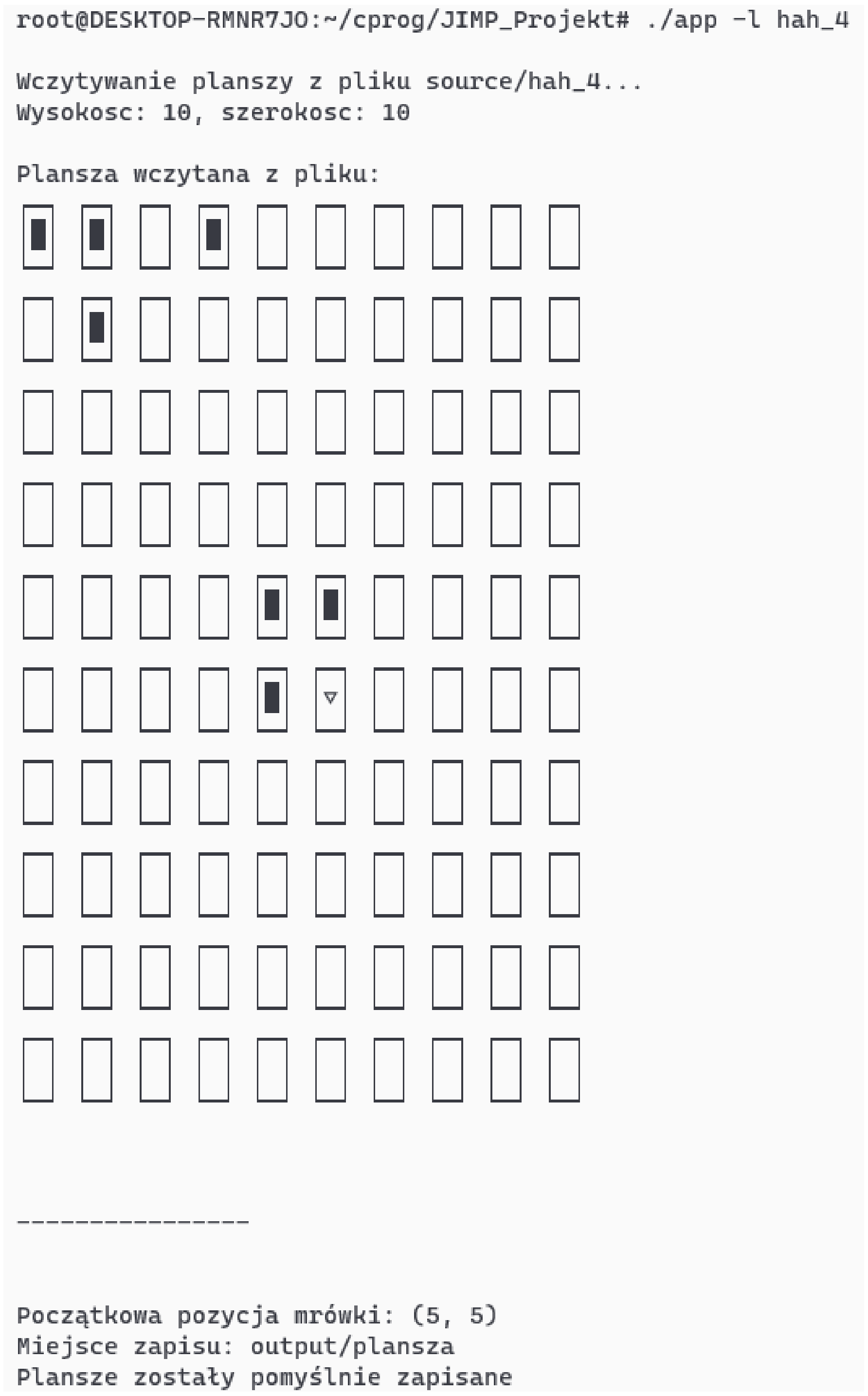
- MODULE **logic**
  The module contains only one function:
  - movement – analyzes the current position of the ant and changes it, in accordance with the assumptions of the Langton Ant. If the ant goes outside the board, the function returns -1.

- MODULE **arguments**
  It contains a structure that stores the properties of the board set by the user.
  It consists of the following functions:
  - help – prints the help menu
  - parseArguments – using the getopt function, analyzes the arguments entered by the user and enters them into the board properties.
- MODUŁ **main**

It calls each of the above modules and simulates the operation of the Langton Ant

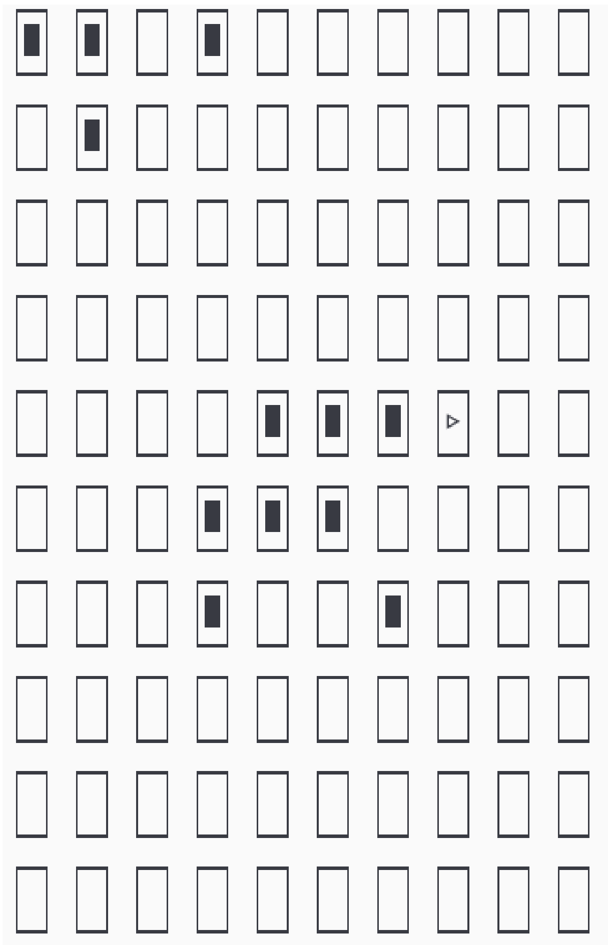
Example Execution

- Loading a board from a file



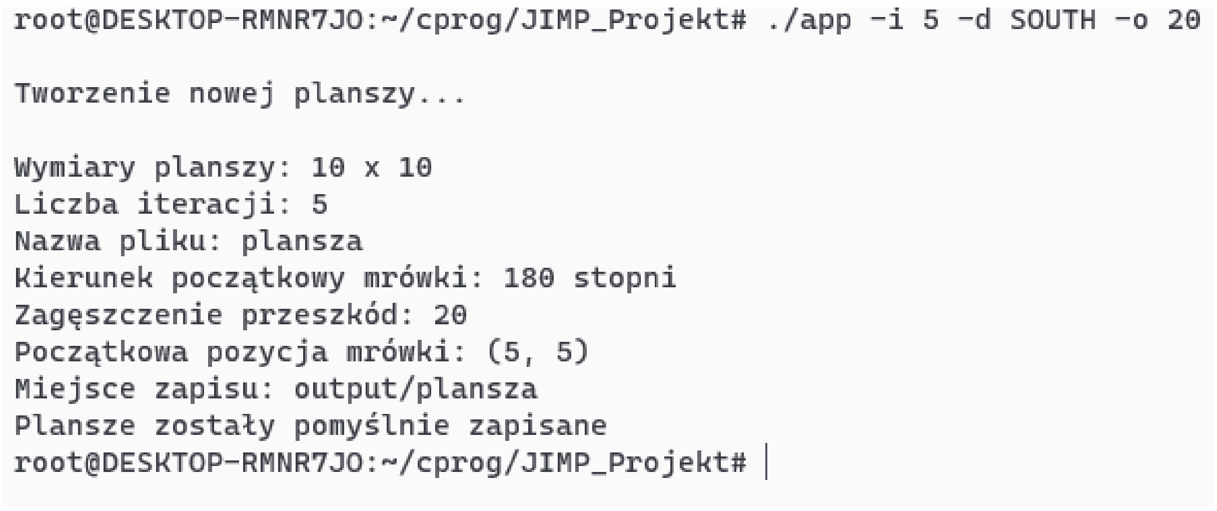
```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# ./app -l hah_4

Wczytywanie planszy z pliku source/hah_4...
Wysokosc: 10, szerokosc: 10

Plansza wczytana z pliku:
▮ ▮ ▯ ▮ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▮ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▮ ▮ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▮ ▽ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯

▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯



----------------


Początkowa pozycja mrówki: (5, 5)
Miejsce zapisu: output/plansza
Plansze zostały pomyślnie zapisane
```

The appearance of the board after 15 iterations:



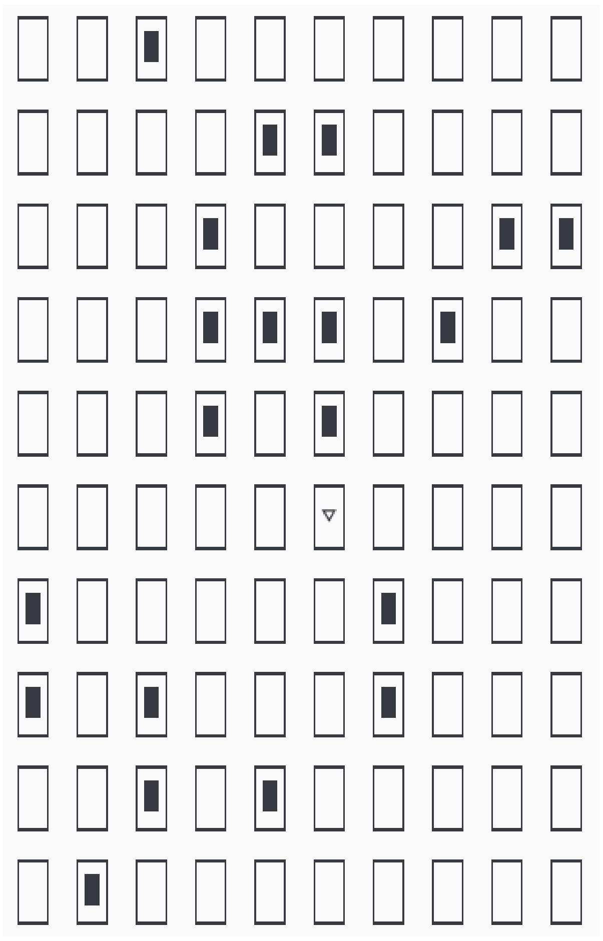
```
▮ ▮ ▯ ▮ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▮ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▮ ▮ ▮ ▷ ▯ ▯
▯ ▯ ▯ ▮ ▮ ▮ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▮ ▯ ▯ ▮ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯ ▯
```

- Changing the number of iterations, starting direction, and adding obstacles:

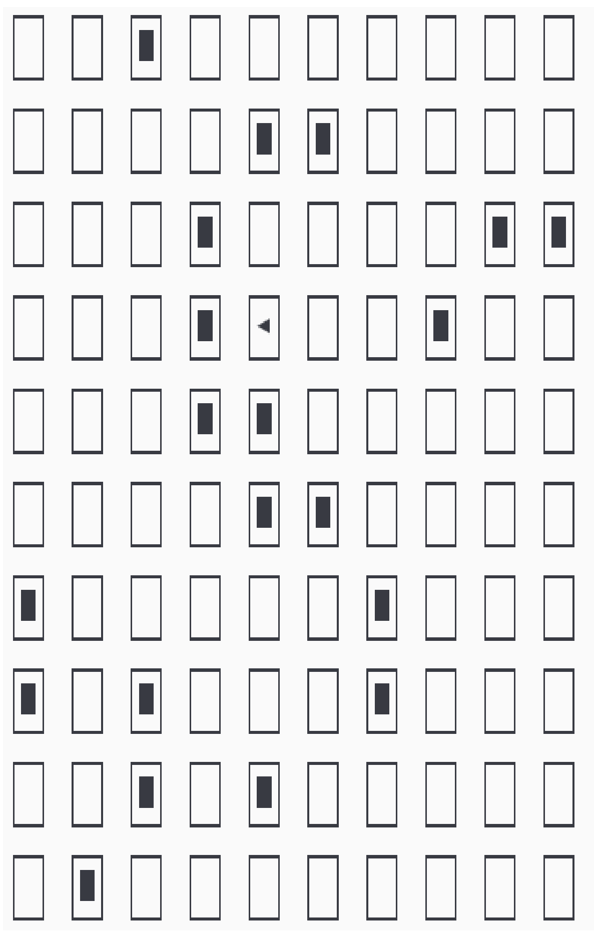


```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# ./app -i 5 -d SOUTH -o 20

Tworzenie nowej planszy...

Wymiary planszy: 10 x 10
Liczba iteracji: 5
Nazwa pliku: plansza
Kierunek początkowy mrówki: 180 stopni
Zagęszczenie przeszkód: 20
Początkowa pozycja mrówki: (5, 5)
Miejsce zapisu: output/plansza
Plansze zostały pomyślnie zapisane
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt#
```

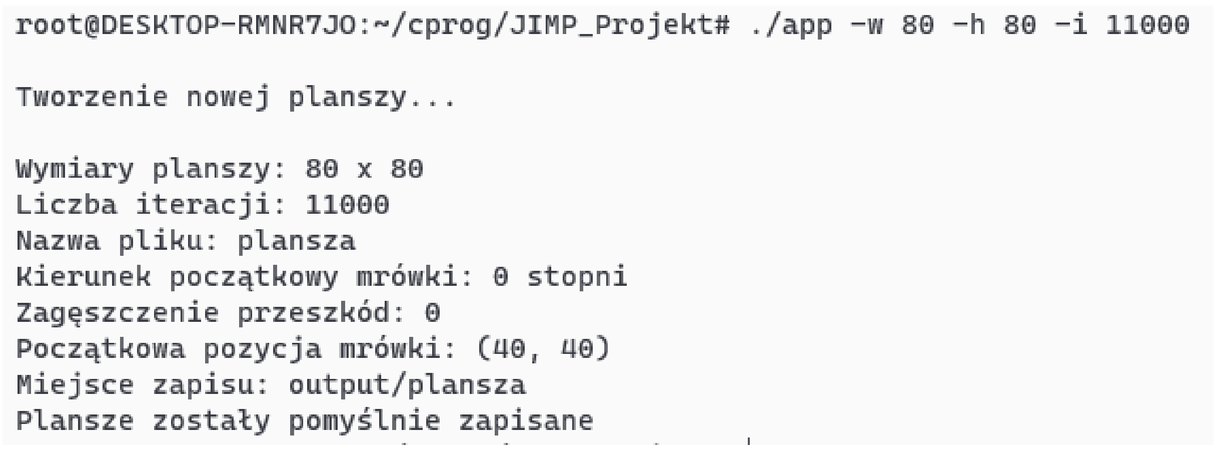
Starting position:



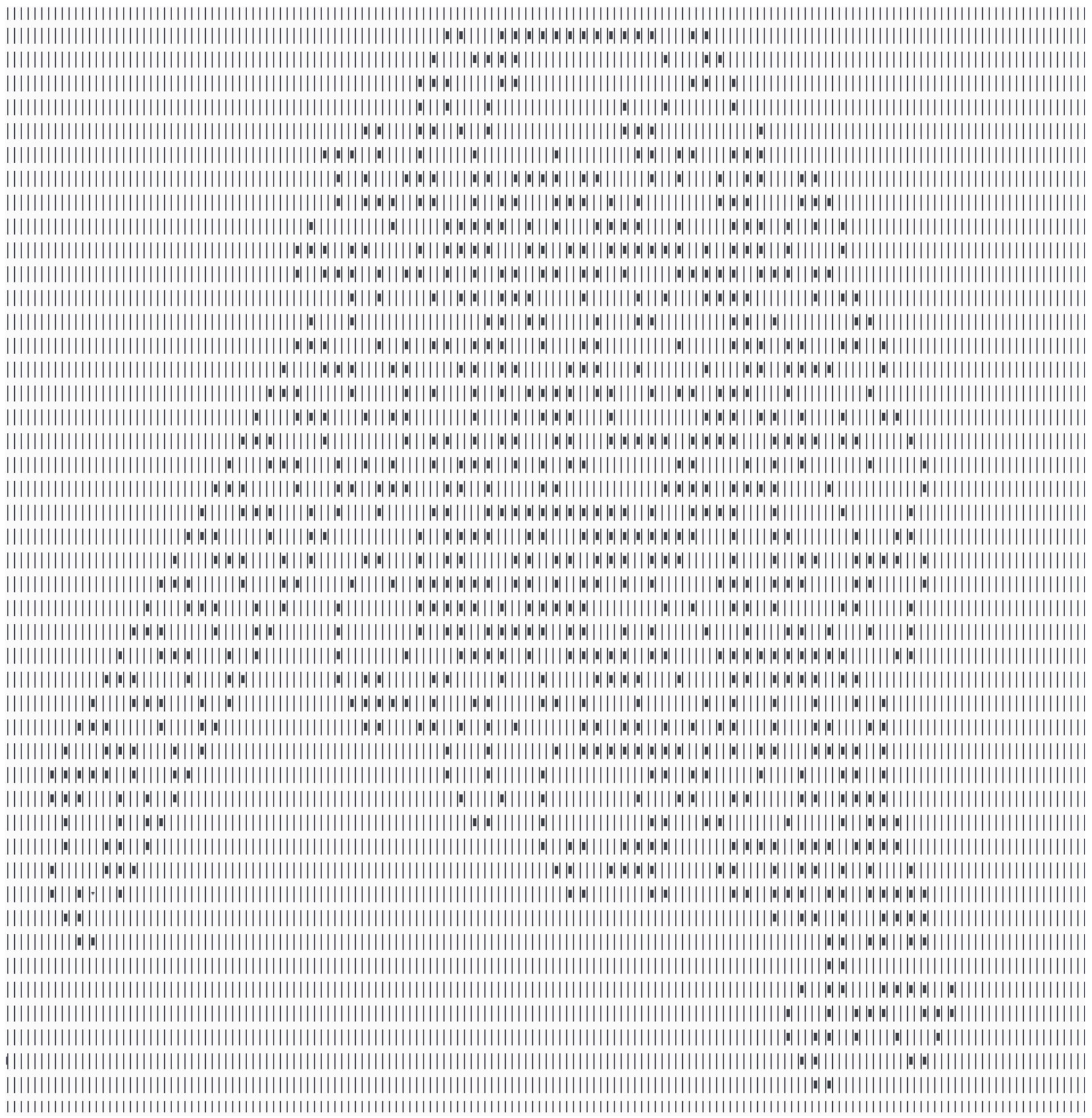
After 5 iterations:



- Ant after 11000 iterations



```
root@DESKTOP-RMNR7JO:~/cprog/JIMP_Projekt# ./app -w 80 -h 80 -i 11000

Tworzenie nowej planszy...

Wymiary planszy: 80 x 80
Liczba iteracji: 11000
Nazwa pliku: plansza
Kierunek początkowy mrówki: 0 stopni
Zagęszczenie przeszkód: 0
Początkowa pozycja mrówki: (40, 40)
Miejsce zapisu: output/plansza
Plansze zostały pomyślnie zapisane
```



Conclusion

Working together on the Langton ant project in C and using GitHub, it was easy to collaborate and share code. By allowing everyone to work on their part of the project at the same time, doing things was more efficient. The project not only helped us learn C programming, but also taught us how to work together effectively using code change tracking tools, which could be helpful for future projects.

Authors: Stanisław Dutkiewicz Filip Kobus