Lab Test Week 9

Examination Rules

This is an exam. Please:

- Wait to be seated by a Lab. supervisor.
- Do not talk to anyone else.
- After logging onto Blackboard (possibly using two-factor authentication) and downloading any files and documents required :
  - Do not use a Web browser, unless instructed to do so (e.g. to submit files).
  - Do not use an email connection.
  - Do not use other electronic devices such as laptops, tablets or phones.
  - Do not look in other peoples directories.
  - Do not use code written in 'groups' with others students.
- **DO** use text books, lecture notes, man pages etc.

If you do not obey the above rules you will be asked to leave.
People arriving late may not be allowed into the Lab.
You will be informed as to the time available. There is additional time in case a server crashes or some other problem occurs.

Submitting Your Work

Submit the `conceal.c` and `inkblot.c` files completed online using the Blackboard submission system. Do not submit files that don't work. Do not submit any driver files provided by me (e.g. `main1.c`, `main2.c`).
We only get to mark your last submission on Blackboard, so **every** time you submit, give us **all** files you want marking.
No partial marks are available. The only possible scores for each sub-part are pass (100%) or fail (0%).

Code Style

Your code **must** compile without warnings using the clang/gcc flags, e.g. :

```
clang main.c file.c -o myprog -Wall -Wextra -pedantic -std=c99 -Wfloat-equal -Wvla -O2
```

You will also need to ennsure your code works with the sanitizer flags :

```
clang main.c file.c -o myprog_s -Wall -Wextra -pedantic -std=c99 -Wfloat-equal -Wvla
-fsanitize=address -fsanitize=undefined -g3
```

Part 1 (60%) : Digit Conceal

As part of an online banking application, you are asked to write a function to partially conceal a users bank account details. If their account number is :

```
1234-567-8901
```

it will be transformed by concealing any digits other than the last four, using the '*' character.

```
****-***-8901
```

Only digits are concealed.
The driver file `main1.c` is given and cannot be changed. You should complete `conceal.c` making any changes that you like, such that both files can be compiled together (and run correctly) without warnings.
A `Makefile` is provided that may be useful.

Part 2 (40%) : Maze Inkblotting

Escaping from a maze can be done in several ways (recursion, righthand-on-wall etc.) but here we look at 'ink blotting'.
The concept is that we mark the entrance to the maze with a particular number. Since 1*s* and 2*s* are used for the walls and spaces, we start with 3 :

2	2	2	2	2	2	2
3	1	2	1	1	1	2
2	1	2	1	2	1	2
2	1	1	1	2	1	2
2	2	1	2	2	1	2
2	1	1	2	1	1	1
2	2	2	2	2	2	2

Then we loop through **every** space in the maze, and set anything adjacent to a 3 as a 4 :

2	2	2	2	2	2	2
3	4	2	1	1	1	2
2	1	2	1	2	1	2
2	1	1	1	2	1	2
2	2	1	2	2	1	2
2	1	1	2	1	1	1
2	2	2	2	2	2	2

We once again loop through **every** space in the maze, and set anything adjacent to a 4 as a 5 :

2	2	2	2	2	2	2
3	4	2	1	1	1	2
2	5	2	1	2	1	2
2	1	1	1	2	1	2
2	2	1	2	2	1	2
2	1	1	2	1	1	1
2	2	2	2	2	2	2

This process is repeated, until the exit square has some number other than 1 in it. After a total of 15 moves, the maze may look something like :

2	2	2	2	2	2	2
3	4	2	10	11	12	2
2	5	2	9	2	13	2
2	6	7	8	2	14	2
2	2	8	2	2	15	2
2	10	9	2	17	16	17
2	2	2	2	2	2	2

and the exit cell has been filled. We can compute the minimum number of moves required to solve the maze using this final number and the initial number.
The driver file `main2.c` is given and cannot be changed. You should complete `inkbot.c` making any changes that you like, such that both files can be compiled together (and run correctly) without warnings.
A `Makefile` is provided that may be useful.
Do not concern yourself with 'weird' edge cases - the maze used will be well formed and have a path from start to exit.