

CS 510: Assignment 1

Due: September 19, 11:55pm

1 Assignment Policies

Collaboration Policy. It is acceptable for students to collaborate in understanding the material but not in solving the problems or programming. Use of the Internet is allowed, but should not include searching for existing solutions.

Under absolutely no circumstances code can be exchanged between students. Excerpts of code presented in class can be used.

Assignments from previous offerings of the course must not be re-used. Violations will be penalized appropriately.

2 Assignment


This project seeks to help you obtain some practice with lists and tuples. A Mini-Logo program is just a list of instructions (integers). The encoding of Mini-Logo instructions is as follows:

Encoding	Instruction
0	Pen down
1	Pen up
2	Move North
3	Move East
4	Move South
5	Move West

An example of a program that draws a square:

[0; 2; 3; 4; 5; 1]

This other one draws two concentric squares

[0; 2; 2; 2; 2; 3; 3; 3; 3; 4; 4; 4; 4; 5; 5; 5; 5; 2; 2; 2; 2; 2; 2; 2; 2; 3; 3; 3; 3; 3; 3;
 3; 3; 4; 4; 4; 4; 4; 4; 4; 4; 5; 5; 5; 5; 5; 5; 5; 5; 1]

This last one draws the letter 'E':

[0;2;2;3;3;5;5;4;3;5;4;3;3;5;5;1]

2.1 Encoding Mini-Logo Programs

We will use the following user defined datatypes.

```
1 type program = int list
```

For example:

```
1 let square : program = [0; 2; 2; 3; 3; 4; 4; 5; 5; 1]
2 let letter_e : program = [0;2;2;3;3;5;5;4;3;5;4;3;3;5;5;1]
```

declares square to be a program.

2.2 Exercises

1. Implement a function

```
colored : int*int -> program -> (int*int) list
```

that given a starting coordinate and a program returns the list of coordinates that the program has colored using its pen. You may introduce helper functions to make your code more readable. Also, you must remove repeated elements from the list.

2. Implement a function

```
equivalent : program -> program -> bool
```

that checks whether two programs are equivalent. We say that two programs are equivalent if they color the same set of coordinates. For example, square above is equivalent to [0; 3; 3; 2; 2; 5; 5; 4; 4; 1].

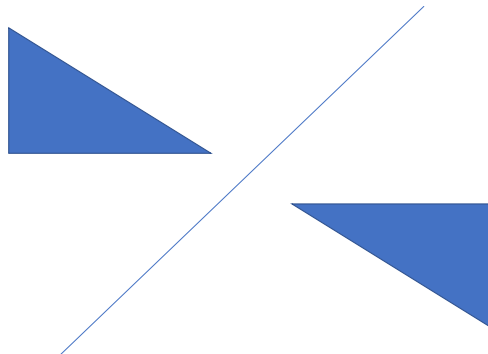
3. Implement a function

```
mirror_image : program -> program
```

that returns a program that draws the mirror image of the input program. For example

```
1 # mirror_image letter_e;;
2 - : program = [0; 4; 4; 5; 5; 3; 3; 2; 5; 3; 2; 5; 5; 3; 3; 1]
```

Hint: use map.



4. Implement a function

```
rotate_90 : program -> program
```

that given a program returns a new one which draws the same pictures except that they are rotated 90 degrees. For example:

```
1 # rotate_image_90 letter_e;;
2 - : program list = [0; 3; 3; 4; 4; 2; 2; 5; 4; 2; 5; 4; 4; 2; 2; 1]
```

Hint: use map.

5. Implement a function

```
repeat : int -> 'a -> 'a list
```

such that repeat n x returns a list with n copies of x . For example:

```
1 # repeat 3 "hello" ;;
2 - : string list = ["hello"; "hello"; "hello"]
```

6. Implement a function

```
pantograph : int -> program -> program
```

such that pantograph p n returns a program that draws the same things as p only enlarged n -fold. For example:

```
1 # pantograph 2 letter_e;;
2 - : program list =
3 [0; 2; 2; 2; 2; 3; 3; 3; 3; 5; 5; 5; 5; 4; 4; 3; 3; 5; 5; 4; 4; 3; 3; 3;
   ↪ 5; 5; 5; 1]
```

Propose three solutions:

- (a) pantograph using explicit matching (i.e. `match`)
- (b) pantograph_m using map; and
- (c) pantograph_f without using map (using fold).

7. Implement a function

```
compress : program -> (int*int) list
```

that compresses a program by replacing adjacent copies of the same instruction with a tuple (m,n) where m is the instruction and n is the number of consecutive times it should be executed. For example,

```
1 # compress letter_e;;
2 - : (int * int) list =
3 [(0, 1); (2, 2); (3, 2); (5, 2); (4, 1); (3, 1); (5, 1); (4, 1); (3, 2)
   ↪ ;(5, 2); (1, 1)]
```

8. Implement a function

```
uncompress : (int*int) list -> program
```

that decompresses a compressed program. For example,

```
1 # uncompress (compress letter_e);;  
2 - : program = [0; 2; 2; 3; 3; 5; 5; 4; 3; 5; 4; 3; 3; 5; 5; 1]
```

Propose three solutions:

- (a) `uncompress_m` using explicit recursion;
- (b) `uncompress_m` using `map`; and
- (c) `uncompress_f` using `fold`.

3 Submission instructions

Submit a single file named `hw1.ml` through Canvas. No report is required. Your grade will be determined as follows:

- You will get 0 points if your code does not compile.
- Partial credit may be given for style, comments and readability.