CISS445: Programming Languages
Test 01 (Written)

Name: ____

This is a closed-book, no-calculator, no-computer, no-discussion test.

Your solution must be written in the box provided. Anything outside the box is not considered part of the solution. Everything inside the box IS considered part of the solution.

Do NOT provide multiple solutions. If you do, I get to pick ONE to grade. (I'm very good at picking solution – the wrong one).

Write neatly. If I cannot read your solution easily you will get zero.

For the OCAML coding problems, your grade will be based on passing a lists of tests cases. If your code passes all the test cases, you get all the points for that question; if it does not pass any test cases you will get 0. Not all test cases are shown below.

Cheating is a serious academic offense. If caught you will receive an immediate score of -100%.

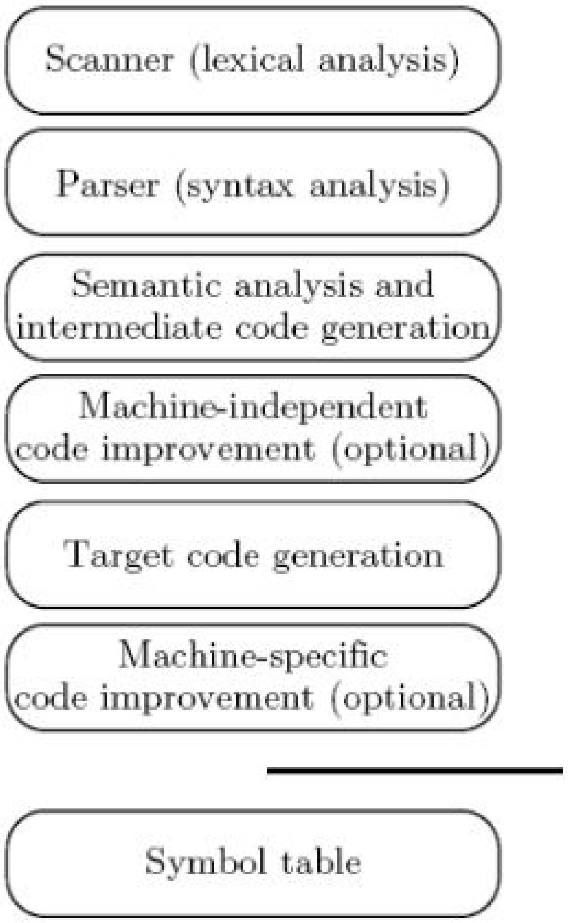
Question	Points
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	

Question	Points
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	
31	
32	
33	
34	
35	
36	
37	
38	
39	
40	

TOTAL	

Q1.

The following are the phases in source code compilation:


Scanner (lexical analysis)
Parser (syntax analysis)
Semantic analysis and intermediate code generation
Machine-independent code improvement (optional)
Target code generation
Machine-specific code improvement (optional)
Symbol table

(a) What is the input for the scanner phase?
SOLUTION.

(b) What is the output during the scanner phase?
SOLUTION.

(c) What is the input for the parser phase?
SOLUTION.

(d) What is the output during the parser phase?
SOLUTION.

Q2.

Write a function `quadratic` such that `(quadratic a b c)` returns a function that computes the y-value of the straight quadratic equation $y = ax^2 + bx + c$. For instance

```
Test                       Expected value
(quadratic 0 1 2) 5        0 * 5 * 5 + 1 * 5 + 2
(quadratic 1 2 3) 3        1 * 3 * 3 + 2 * 3 + 3
(quadratic 2 3 4) 7        2 * 7 * 7 + 3 * 7 + 4
```

SOLUTION.

Q3.

Write a `head` function that returns the first value of a list. Ignore the case where the list is empty.

```
Test            Expected value
head [1]        1
head [2;9]      2
head [3;2;5]    3
```

SOLUTION

Q4. What is the value of the x:

```
let x = (fun f -> (fun x -> f(f x))) (fun x -> x * x) 2;;
```

or write ERROR

SOLUTION

Q5.

Write down the type for each of the following expressions.

- Recall that the type of a function look like `'a -> 'b`. For instance the type of `fun x -> x + 1` is `int -> int` and the type of `fun x -> fun y -> x + y` is `int -> int -> int` and the type of `fun x -> fun y -> (x < y)` is `'a -> 'a -> bool`.
- The type of a list looks like `'a list`. For instance the type of `[1;2;3]` is `int list`.
- The type of a 2-tuple looks like `'a * 'b`. For instance the type of `(1, 2.4)` is `int * float`. The type for 3-tuple, 4-tuple, etc. are analgous.

(a) `if 1 = 2 then [3] else [4]`
SOLUTION

(b) `fun x -> fun y -> y::x::[1]`
SOLUTION

(c) `fun x -> fun y -> y::[x]`
SOLUTION

(d) `fun x -> fun y -> (y ((x + 1) + 2) + 3.0`
SOLUTION

Q6.

Write a number guessing game. Call `(guess 42)` and the user is prompted to enter a guess. If the number entered is 42, the program will stop. If the number entered is < 42, the program will ask the user to try a larger number. If the number entered is > 42, the program will ask the user to try a smaller number.

To get an integer value from the user do this:

```
let x = read_int ();;
```

Here's a sample execute:

```
utop # guess 42;;
guess my int: 1
try higher ...
guess my int: 100
try lower ...
guess my int: 50
try lower ...
guess my int: 25
try higher ...
guess my int: 30
try higher ...
guess my int: 40
try higher ...
guess my int: 45
try lower ...
guess my int: 43
try lower ...
guess my int: 42
correct!!!
- : unit = ()
```

Turn page ...

SOLUTION

Q7.

Write a function `reverse_at` so that `(reverse_at list n)` returns the element of the list at index n in the order, i.e., index 0 means the last value, index 1 means the second last value, etc. Ignore the case where `n<0` or `n>=` the size of the list.

Test	Expected values
`reverse_at [5;3;1] 0`	`1`
`reverse_at [2;4;6] 1`	`4`
`reverse_at [1;3;5;7] 3`	`1`

SOLUTION

Q8. Write a function `rev` that reverses a list. For instance after

```
let xs = rev [1;3;5];;
```

`xs` is `[5;3;1]`. Any recursion used must be tail recursion.

SOLUTION