UCLA CS 131 Midterm, Winter 2010
100 minutes total, open book, open notes

Name: Jake Stothard Student ID: 903645927

1	2	3	4	5	6	7	8	9
21	9	7	8	15	10	3	2	8

10	11	12	sum
6	4	5	99

1. What are the types of the functions f, g, h defined by the following OCaml declarations? For each function, give an example call to that function.

a (5 minutes).
```
let f x = x []
```

b (7 minutes).
```
let g a b c = a b c
```

c (12 minutes).
```
let rec h m n ls = match ls with
  []  -> n
  x::t -> h m (m n x) t
```

2 (10 minutes). Is the following definition equivalent in type and behavior to the definition in (1c)? If so, explain why; if not, explain why not.

```
let h m =
  let rec h1 n = function
    []  -> n
    x::t -> h1 (m n x) t
  in h1
```

3. Consider the following code, taken from the hint to Homework 2:

```
let rec match_star matcher frag accept =
  match accept frag with
    | None ->
        matcher frag
          (fun frag1 ->
            if frag == frag1
            then None
            else match_star matcher frag1 accept)
    | ok -> ok
```

What would happen if you did the following? (Consider each change independently.)

3a (3 minutes). Omit the "rec".

3b (3 minutes). Delete the last line of this function.

3c (3 minutes). Replace the last line with "|Some x -> Some x".

3d (3 minutes). Move the last line so that it is between the second and third lines.

3e (3 minutes). Replace the if-then-else expression with its then-part "None".

3f (5 minutes). Replace the if-then-else expression with its else-part "match_star matcher frag1 accept".

4a (4 minutes). Give a specific example grammar that will make a good solution to Homework 1 go into an infinite loop.

4b (6 minutes). Give a specific example grammar that will make a good solution to Homework 2 take a very long time to finish, though it won't loop forever.

5a (12 minutes). Write an OCaml function "check_perm A B" that returns true if A is a permutation of B. A and B must be lists. Don't worry about efficiency, though your function must terminate reliably. For example, 'check_perm [1;3;2] [2;1;3]' should return true, and 'check_perm ["a";"b"] ["a"]' should return false. It's OK to define some auxiliary functions in order to implement check_perm.

5b (3 minutes). What is the type of check_perm?

6 (12 minutes). Suppose you are assigned the task of solving Homework 3 in the functional subset of OCaml. This subset doesn't have threads, so how would you reformulate the tasks of Homework 3 (such as matching halves of the input file separately) while retaining the spirit of the problem? Illustrate your ideas with some OCaml code.

7 (12 minutes). Suppose the only form of synchronization in Java was the "synchronized" keyword. Explain how you would have solved the problem in Homework 3, using only this form of synchronization. Compare the quality of the solution with "synchronized" versus the solution you adopted in your answer to Homework 3.

8a (10 minutes). Suppose that Java had been designed with a class Float (instead of with a primitive type 'float'), with subclasses Normalized, Tiny, Infinite, and NaN for each of the major kinds of IEEE-754 floating-point numbers. Give signatures for some example methods for this type system, and show how to calculate the value of the expression (d == b*b - 4*a*c) using these methods, where a, b, c, and d are of type Float. Give particular consideration to exception handling and to mixed-mode computation (where some of the operands are int instead of Float).

8b (5 minutes). In (8a), would it make more sense to have Float be an interface than a class? How about an abstract class? Explain.

9. Consider the following grammar for msg-id, modified from Internet RFC 2822:

```
msg-id          = "<" id-left "@" id-right ">"
id-left         = dot-atom-text
id-right        = dot-atom-text / no-fold-literal
no-fold-literal = msg-id
dot-atom-text   = 1*atext *("." 1*atext)
atext           = "a" / "b" / "c" / "d" / "@"
```

9a (8 minutes). Prove that this modified grammar is ambiguous.

9b (8 minutes). Modify the grammar to make it unambiguous. Make as few changes to the grammar as you can.

9c (9 minutes). Give a syntax diagram for the unmodified grammar.

10 (15 minutes). In class we talked about a software-tools approach to translation, in which translation from source to machine code is divided into several steps. Suppose we want to do decompilation, that is, we want to translate from machine code that is the output of a compiler, back to the source code that generated this machine code. Which of the software-tools steps would be the most difficult, and why? Which would be easier, and why?

11 (12 minutes). Would it make sense to modify Java so that it supported type inference a la OCaml? The idea would be to make upwards-compatible changes to Java, so that programmers don't have to write down the types if they don't want to. Assume that you can add a few new keywords, if that would help make the changes. Justify your answer.

12 (10 minutes). Suppose I design a new language Noreturn-Java. Noreturn-Java is just like Java, except that there's no "return" keyword. To return the the value of an expression E, simply write "E;", without writing "return". For example, the Fibonacci function can be written as follows:

```
public static long fib(int n) {
    if (n <= 1) n;
    else fib(n-1) + fib(n-2);
}
```

Other than the usual backwards compatibility issues, what major problem do you see when implementing Noreturn-Java? Give a specific example of the problem.

Jake Stothard
903645

1. a) ('a list → 'b) → 'c = <fun>

3. f (fun a -> if (a = []) then "yes" else "no")

b) ('a → 'b → 'c) → 'a → 'b → 'c = <fun>

2\ 7 g (fun x y = "hi") "ignore1" 5

c) ('a →(m) 'b → 'a) → 'a →(n) 'b list → 'a = <fun>

11. h (fun a b → "cake") "sleep" [1;2;3]

2. yes, it is the same in type and behavior. Since m never changes when recursively calling, it is the same to bring it out. The keyword function is the same as the code "ls = match ls with" here.

3. a) The line "else match-star..." would be a compile time error and it would call it an Unbound value.

3 b) There would be a warning that the pattern matching is not exhaustive, but it would still compile. If accept frag ever was ok it would cause a runtime error.

9 c)

0 d) There would be no change

1 e) The acceptor passed to the matcher would now accept nothing. This matcher should now return None.

0 f) There would be no case in which match_star returns None.

4

4. a) $S \to S$

```
let my_nonterm = S
(S, function | S -> [[N S]])
```

b)
$A \to aBC$
$B \to bB$
$C \to cA$
$A \to a$
$B \to b$

```
let my_nonterm2 = A | B | C
(A, function | A -> [[T "a"; N B; N C]; [T "a"]]
             | B -> [[T "b"; N B]; [T "b"]]
             | C -> [[T "c"; N A]])
```

4

5. a)

```
Let rec check_perm a b =
  let rec check_length a b = match a with
      | [] -> if (b = []) then true else false
      | h::t -> match b with
          | [] -> false
          | h2::t2 -> check_length t t2 in
  let rec remove_elem ls a = match ls with
      | [] -> []
      | h::t -> if (h = a) then t else h::(remove_elem t a) in
  if (check_length a b) = false then false else match a with
      | [] -> true
      | h::t -> check_perm t (remove_elem b h)
```

12

5. b) 'a list → 'a list → bool

6.

```
Let grep2 pattern filestring =
(*Ocaml can run the parts in parallel automatically once you
divide it into two parts*)
  let grep pattern filestring = ??? in
  (* Assume grep is given as it was in homework 3 *)
  let finddiver string startpos = in
  (* find the middle \n *)
  let divide string divider = in (*returns tuple of first and second
                                   part of string
  let (firstpart, secondpart) = divide filestring (finddivider filestring (sz/2)) in
  (grep firstpart) @ (grep second part)
  (*Assume grep returns a list of its results. there are
```

10

Jake
Stat
903645

7. If Java only had the synchronized keyword, I would create an Object to be the lock on the output. The first thread holds this until it is done with it's output. The once the second thread is done finding matches, it waits until the synchronized object is released by the first thread to start outputting its results. Although this is as fast as my solution, it could not match code which began outputting the caters stored by thread 2 once thread 1 is finished even if thread 2 is still putting matches in the queue. This would be possible with an exchanger.

? don't see how it'd work

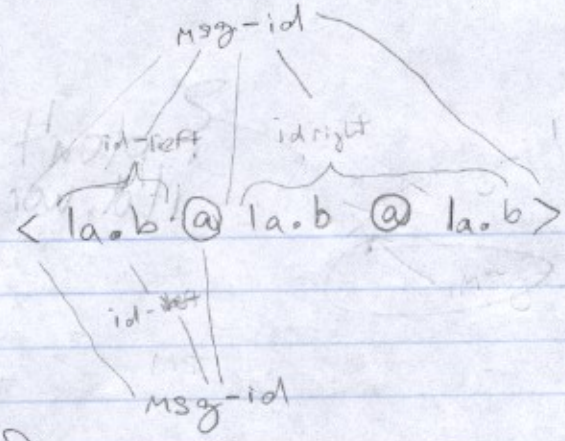
that's synchronization

8. a) /* return whether Float is Normized, Tiny, etc. Number to represent each
public int getType() { return myType }
*/ public

b) It would not make as an interface since it has some methods such as addition which it should have defined. It would not make sense as an abstract class because then Float a; would not be allowed.

yes, Float a; would be fine. You can say a = new Tiny(...);

what is bad about that?



9. a)

8

10. Translating from machine code to assembly (as) is easy ✓ since it is almost a direct translation. Undoing the linking step (ld) would require finding which code is contained in other libraries so would be more difficult, but if this step fails the source would just have extra code. Undoing the compilation would require guesswork to undo all the optimizations and put the source in a form readable to people, making it the most difficult step. ✓

6

11. Some type inference could be done in Java, but the option to put the type name must still be there. Imagine: ✓

```
public class Person { public String a() { return "hello"; } }
public class Person2 extends Person { public String a() { return "hi"; }
public static void main (String[] args) {
    Person p = new Person2();
    System.out.println(p.a());
}
```

4

Relying on any type inference here would probably not get the same behavior. As things got more complicated this difficulty would become a lot more apparent.

Don't see the difference in this example.

Jake
Stothard
903645927

12. Since Java has statements, it would be unclear when to return.

```
public static int test(int a) {
    int w;
    if (a > 0)
        w = a+1;   <- Do we want to return here?
    else
        w = a-1;
    q = w+5;
}
```

Each assignment in Java returns a value so that statements like a=b=c; work. Here, it is unclear whether the code should return the value found in the first part of the if or go to the last line. Something could be defined for how it will behave, but then it makes it more complicated for the programmer.

You can't say
(return w=a+1;)
in Java,