Please do not turn the page until 12:30.

Rules:

- The exam is closed-book, closed-note, except for one side of one 8.5x11in piece of paper.
- **Please stop promptly at 1:20.**
- You can rip apart the pages, but please staple them back together before you leave.
- There are **100 points** total, distributed **unevenly** among **5** questions (all with multiple parts).
- When writing code, style matters, but don’t worry much about indentation.

Advice:

- Read questions carefully. Understand a question before you start writing.
- Write down thoughts and intermediate steps so you can get partial credit.
- The questions are not necessarily in order of difficulty. **Skip around.** Make sure you get to all the problems.
- If you have questions, ask.
- Relax. You are here to learn.
1. This problem uses this datatype binding, where a value of type \texttt{points} describes a set of points on the plane, i.e., a 2-D plot with an \texttt{x}-axis and a \texttt{y}-axis.

\begin{verbatim}
datatype points = Point of real * real
  | Seg of real * real * real * real
  | Union of points * points
  | Shift of points * real * real

• \texttt{Point(x,y)} represents the point \texttt{(x,y)}.
• \texttt{Seg(x1,y1,x2,y2)} represents all points on the line segment with endpoints \texttt{(x1,y1)} and \texttt{(x2,y2)}.
• \texttt{Union(s1,s2)} represents all points represented by \texttt{s1} unioned with all points represented by \texttt{s2}.
• \texttt{Shift(s,dx,dy)} represents the points represented by \texttt{s} after shifting them to the right by \texttt{dx} and up by \texttt{dy}.
\end{verbatim}

Note: we did not use type \texttt{real} much in class, but you can use arithmetic operations (e.g., +) and comparison operations (e.g., >) as expected.

(a) (12 points) Write an ML function \texttt{rightmost} of type \texttt{points -> real * real} such that \texttt{rightmost s} returns the point in the set represented by \texttt{s} with the largest \texttt{x}-coordinate. (You can resolve ties however you wish.) Notice the result type is \texttt{real * real}, the \texttt{x}-coordinate and \texttt{y}-coordinate.

(b) (12 points) Write an ML function \texttt{max_shifts} of type \texttt{points -> int} that given \texttt{s} computes the maximum number of shifts that apply to a single “point” or “segment” in \texttt{s}. Note this is \textit{not} necessarily the number of \texttt{Shift} constructors in \texttt{s}. For example, the correct answer for

\begin{verbatim}
Union(Shift(Point(0.0,0.0),1.0,1.0),
  Shift(Union(Shift(Point(2.0,2.0),1.0,1.0),
    Shift(Shift(Seg(3.0,4.0,5.0,6.0),7.0,8.0),9.0,10.0)),
  20.0,75.0))
\end{verbatim}

is 3 because the one segment is under three \texttt{Shift} constructors, including the one outside the nested \texttt{Union}.
Name: ____________________________________________

More room for Problem 1 in case you need it
2. This problem uses these two similar but different functions:

```
fun f1 (xs,ys) =
  case (xs,ys) of
    ([], []) => []
  | (x::xs', y::ys') => (x,y)::(f1(xs',ys'))
  | (x::xs', []) => []
  | ([], y::ys') => []

fun f2 (xs,ys) =
  case (xs,ys) of
    ([],[]) => []
  | (x::xs', y::ys') => (x,y)::(f2(xs',ys'))
  | (x::xs', []) => (x,0)::(f2(xs',[]))
  | ([], y::ys') => (0,y)::(f2([],ys'))
```

(a) (5 points) Fill in the blanks so that \(c_1\) and \(d_1\) are both bound to \([(2,2),(1,1),(0,0)]\)

\[
\begin{align*}
\text{val } a1 &= \underline{\text{------------------}} \\
\text{val } b1 &= \underline{\text{------------------}} \\
\text{val } c1 &= f1(a1,b1) \\
\text{val } d1 &= f2(a1,b1)
\end{align*}
\]

(b) (5 points) Fill in the blanks so that \(d_2\) but not \(c_2\) is bound to \([(2,2),(1,1),(0,0)]\)

\[
\begin{align*}
\text{val } a2 &= \underline{\text{------------------}} \\
\text{val } b2 &= \underline{\text{------------------}} \\
\text{val } c2 &= f1(a2,b2) \\
\text{val } d2 &= f2(a2,b2)
\end{align*}
\]

(c) (5 points) Fill in the blanks so that \(c_3\) but not \(d_3\) is bound to \([(2,2),(1,1),(0,0)]\)

\[
\begin{align*}
\text{val } a3 &= \underline{\text{------------------}} \\
\text{val } b3 &= \underline{\text{------------------}} \\
\text{val } c3 &= f1(a3,b3) \\
\text{val } d3 &= f2(a3,b3)
\end{align*}
\]
3. For each of the following programs, give the value that `ans` is bound to after evaluation:

(a) (4 points)

```plaintext
val x = 1
fun f y = 
    let
        val x = y + 1
        val y = x + 1
    in
        y + 1
    end
val z = f 4
fun f x = x
val ans = z
```

(b) (4 points)

```plaintext
val x = 1
val y = 2
fun f (g,h) = g x + h y
val x = 3
val y = 4
val ans = f ((fn z => x), (fn z => z))
```

(c) (4 points)

```plaintext
exception E
val x = 1
fun f x = if x=2 then raise E else 14
val x = 2
val ans = ((f x) + 4) handle E => 9
```

(d) (4 points)

```plaintext
val z = 2
val f = (fn x => x + 1) o (fn y => if y=z then 4 else y)
val z = 3
val ans = List.map f [1,2,3,4,5]
```
4. (a) (10 points) Without using any helper functions (such as foldl), write an ML function `in_order` that behaves as follows:

- It takes two arguments *in curried form*: (1) a function \( f \) that given a list element produces an integer and (2) a list \( xs \).
- It returns true if and only if for all elements of \( xs \), \( f \) applied to the element returns a number less than or equal to \( f \) applied to any later elements of the list. (This means the result is true for any list with fewer than two elements.)

(b) (6 points) Using `in_order`, write a function `shorter_strings` that takes a list of strings and returns true if and only if each string in the list is *longer* than the strings that come later in the list. Hint: You can use ML’s \( \sim \) operator for negation.

(c) (4 points) What is the type of `in_order`?

(d) (2 points) What is the type of `shorter_strings`?

(e) (4 points) When your solution to part (a) is given a list \( xs \) of length \( n \), how many times is the function passed for \( f \) called before `in_order` returns?

(f) (3 points) Suppose another student has a different answer to part (e) and you are both correct because you have different correct answers to part (a). Are your solutions to part (a) equivalent? Explain briefly.
5. In this problem, suppose we have an ML structure `M` and signature `S` in this standard usage:

```ml
signature S =
sig
  ...
end
structure M :> S =
struct
  ...
end
```
Assume everything type-checks initially, meaning `M` matches `S`. For each of the following statements, answer “always,” “sometimes,” or “never.”

(16 points) (2 points each)

(a) If `S` originally contains `val f : int -> int` and we comment out this line, then `M` will still match `S`.
(b) If `S` originally contains `val f : int -> int` and we comment out this line, then a client of `M` will still type-check.
(c) If `S` originally does not contain `val g : string -> string` and we add it to `S`, then `M` will still match `S`.
(d) If `S` originally does not contain `val g : string -> string` and we add it to `S`, then a client of `M` will still type-check.
(e) If `S` originally contains an abstract type `type t` and we replace this line with `datatype t = Foo of int | Bar of bool`, then `M` will still match `S`.
(f) If `S` originally contains an abstract type `type t` and we replace this line with `datatype t = Foo of int | Bar of bool`, then a client of `M` will still type-check.
(g) If `S` originally contains the line `datatype t = Foo of int | Bar of bool`, and we replace this line with `type t`, then `M` will still match `S`.
(h) If `S` originally contains the line `datatype t = Foo of int | Bar of bool`, and we replace this line with `type t`, then a client of `M` will still type-check.
Name:______________________________

*More room in case you need it.*