For this assignment, each student will turn in a write-up. **You need to have your own write-up.** As always, you are welcome and encouraged to collaborate and discuss these questions in groups. Just be sure to acknowledge those with which you discussed.

**Submission Instructions**

To submit, upload to the moodle exactly two files named as follows:

- hw4-YourIdentiKey.pdf with your answers to the written questions
- hw4-YourIdentiKey.sml with your answers to the coding exercises

Replace `YourIdentiKey` with your IdentiKey (e.g., for me, I would submit hw4-bec.pdf and hw4-bec.sml). To help with managing the submissions, we ask that you rename your uploaded files in this manner beginning with this assignment.

**Grading Comments**

- Make sure your code compiles and runs on CSEL. Code that does not compile and run will definitely not get full credit.

- Try to make your code as clean, simple, and elegant as possible. Not every working implementation deserves the same credit. None of the coding exercises require more than a few lines of code.

- Document any assumptions you make about the range of inputs to a function (e.g., (* invariant n >= 0 *)).

**Getting Started**

Download the code template hw4.sml from the assignment page.
Exercise 1: Bookkeeping. Indicate in a sentence or two how much time you spent on this homework, how difficult you found it subjectively, and what you found to be the hardest part. Tell me something about yourself that I do not already know. Any non-empty answer will receive full credit.

Also, if your opinions have changed since the last assignment, indicate one thing you like about the class so far and one thing you would change about it.

Exercise 2: Binding and Scope. Skill 7.3. You should solve the parts of this exercise in your head (i.e., without using the SML interpreter).

1. Consider the following SML program:

```
val pi = 3.14
fun circumference r = 
  let
    val pi = 3.14159
    in
    2.0 * pi * r
  end
fun area r = pi * r * r
```

What value is pi bound to at line 6? What value is pi bound to at line 8? Briefly explain your reasoning.

2. Consider the following SML program:

```
val x = 3
fun f 0 = 1
| f x = 
  let
    val x = x + 1
    in
    x
  end
* f (x - 1)
val y = x + f x
```

The use of x on line 5 is bound on which line? The use of x on line 7 is bound on which line? The use of x on line 9 is bound on which line? The use of x on line 10 is bound on which line? Briefly explain your reasoning.
**Exercise 3:** **Skill 7.4.** Given the following declaration:

```sml
fun f 0 = (1, 1)
  | f n =
    let
      val (x, y) = f (n - 2)
    in
      (y, x + y)
    end
```

What is the type and value of the following expression:

```
f 3
```

if they exist? Explain your reasoning in 1-2 sentences.

**Exercise 4:** **Skill 7.1.** In this exercise, you will write a few functions in SML.

1. Implement a function

   ```sml
   concatn : string * int -> string
   ```

   where `concatn(s,n)` returns a string with `n` copies of `s` concatenated together. For example, `concatn("a",3)` returns "aaa".

2. Recall that in SML, functions can they can passed as arguments to or returned from other functions, just like any other value. Implement a function

   ```sml
   foldn : (int -> int) * int * int -> int
   ```

   where `foldn(f,b,n)` applies the function `f` to the base value `b` the number `n` times. In other words, it returns

   \[
   f(f(\ldots f(b)\ldots))
   \]

   \[
   \text{n times}
   \]

3. Recall the direct implementation of the exponentiation function from class:

   ```sml
   fun exp (x, 0) = 1
   | exp (x, n) = x * exp (x, n - 1)
   ```
that computes \( x^n \). For this exercise, provide a non-recursive implementation of the exponentiation function

\[
\text{exp} : \text{int} * \text{int} \rightarrow \text{int}
\]

by calling \text{foldn}. That is, \text{exp} is not itself recursive but uses the recursive function \text{foldn} defined above. For grading purposes, you will not penalized on this part if your \text{foldn} is not working.

**Exercise 5: Skill 7.1.** In this exercise, you will implement a function that approximates definite integrals of the cosine function.

Recall from Calculus that the definite integral of a positive function is the area under the curve in an interval. One way we can approximate a definite integral is to sum many narrow rectangles as follows:

Implement a function

\[
\text{cosrectangles} : \text{real} * \text{real} * \text{int} \rightarrow \text{real}
\]

where \text{cosrectangles}(a, b, n) approximates the cosine function over the interval \([a, b]\) using \( n \) rectangles. Each rectangle should have width

\[
\frac{(b - a)}{n}
\]

and height equal the cosine of the left endpoint. You may assume the following about the arguments to \text{cosrectangles}.

\[\text{Math.pi} \leq a < b \leq \text{Math.pi} \text{ and } n > 0\]
Hints. The cosine function in SML is

\[ \text{Math.cos : real -> real} \]

You can convert ints to reals using

\[ \text{Real.fromInt : int -> real} \]