Writing a very simple SML program

Up until now we've covered equal parts theory and language constructs. . . but we haven't written any code yet.
Remember to think through, and document in comments, your strategy before you start coding!

```
(*
Strategy:
__________
1. Parse input
2. Convert to standard form
3. Do calculations
*)
```

Pattern Matching

- Pattern matching is used extensively in SML.
- It's similar to (but much more powerful than) case or switch statements in other languages.
    - You can also operate on and name the parameters to your case statement or function.

Example – Select even numbers from a list and test the function:

```
fun GrabEvens(nil) = nil
|   GrabEvens(n::ns) =
    if n mod 2 = 0 then
        n::GrabEvens(ns)
    else
        GrabEvens(ns);

GrabEvens([1,2,3,4,5,6,7,8,9,10]);
```

Running from the Command Line: Style 1

- Use the `use “filename”;` command in the SML REPL interpreter to load and execute a file

Example – Loading and running the previous example:

```
- use "GrabEvens.sml";
[opening GrabEvens.sml]
val GrabEvens = fn : int list -> int list
val it = [2,4,6,8,10] : int list
val it = () : unit
```

```
- GrabEvens([21,22,23,24,25]);
val it = [22,24] : int list
```

Running from the Command Line: Style 2

- Use the `sml filename` command in the shell (not the SML interpreter) to load a file into the interpreter and stay in the interpreter after execution

Example – Loading and running the previous example:

```
$ sml GrabEvens.sml
Standard ML of New Jersey v110.78 ...
[opening GrabEvens.sml]
val GrabEvens = fn : int list -> int list
val it = [2,4,6,8,10] : int list
```

```
- GrabEvens([21,22,23,24,25]);
val it = [22,24] : int list
```

Running from the Command Line: Style 3

- Use the `sml < filename` command in the shell to load a file into the interpreter and return to the shell after execution

Example – Loading and running the previous example:

```
$ sml < GrabEvens.sml
Standard ML of New Jersey v110.78 ...
- val GrabEvens = fn : int list -> int list
val it = [2,4,6,8,10] : int list
-
$
```

Lab #1

- Find the sum of all the multiples of 3 or 5 below some given natural number n.

Your program should be executable at the SML prompt by:

```
- use "Lab1.sml";
- Lab1(n);
```

Example inputs and outputs

```
- Lab1(10);
val it = 23 : int
- Lab1(100);
val it = 2318 : int
- Lab1(1738);
val it = 705570 : int
```

– Problem from Project Euler