MP 1 – Basic OCaml

CS 421 – Fall 2014
Revision 1.0

Assigned August 26, 2014
Due September 2, 2014, 23:59 PM
Extension 48 hours (penalty 20% of total points possible)

1 Change Log

1.0 Initial Release.

2 Objectives and Background

The purpose of this MP is to test the student's ability to

- start up and interact with OCaml;
- define a function;
- write code that conforms to the type specified (this includes understanding simple Ocaml types, including functional ones);

Another purpose of MPs in general is to provide a framework to study for the exam. Several of the questions on the exam will appear similar to the MP problems. By the time of the exam, your goal is to be able to solve any of the following problems with pen and paper in less than 2 minutes.

3 What to `commit`

You should put code answering each of the problems below in a file called `mp1.ml`. A good way to start is with the stub file we have given you, editing to remove the stubs and insert you answers. If you choose not to start this way, please be sure to place

```
open Mp1common
```

at the top of you file. Please read the *Guide for Doing MPs* in

`http://courses.engr.illinois.edu/cs421/mps/index.html`

The command to commit this file is:

```
svn commit -m "Turning in mp1." mp1.ml
```

4 Problems

Note: In the problems below, you do not have to begin your definitions in a manner identical to the sample code, which is present solely for guiding you better. However, you have to use the indicated name for your functions and values, and they will have to conform to any type information supplied, and have to yield the same results as any sample executions given, as well as satisfying the specification given in English.

1. (1 pt) Declare a variable `random` with the value `17`. It should have type `int`.

2. (1 pt) Declare a variable `pi` with a value of `3.14159`. It should have the type of `float`.

3. (2 pts) Write a function `myFirstFun` that returns the result of mulitplying the sum of a given integer and 3 by 4.

```
# let myFirstFun n = ... ;;
val myFirstFun : int -> int = <fun>
# myFirstFun 17;;
- : int = 80
```

4. (2 pts) Write a function `circumference` that, when given a radius as a `float`, returns the circumferenceof a circle of that radius. You should use the value given in problem 2 for π.

```
# let circumference r = ...;;
val circumference : float -> float = <fun>
# circumference 1.0;;
- : float = 6.28318
```

(Your value may vary slightly from that printed here if you use a machine of different precision.)

5. (2 pts) Write a function `double` that takes an `int` and returns the pair of `int`s where the first is the number given and the second is two times that.

```
# let double n = (n, 2*n);;
val double : int -> int * int = <fun>
# double 0;;
- : int * int = (0, 0)
```

6. (5 pts) Write a function `make_bigger` that takes a `float` and if it is less than 0.0, multiplies it by (-1.0) (use the parentheses), it is at least 0.0, but less than 1.0, adds 0.5 to it, and if it is at least 1.0, squares it.

```
# let make_bigger x = ...
val make_bigger : float -> float = <fun>
# make_bigger 12.0;;
- : float = 144.
```