

---

# MP 1 – Basic OCaml

CS 421 – Spring 2008

Revision 1.0

**Assigned** January 15, 2008

**Due** January 17, 2008, 11:59 PM

**Extension** 48 hours (penalty 20% of total points possible)

---

## 1 Change Log

1.1 Removed an “h” from the “whachya” of the “salutations” function to make it agree with the grader.

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);
- write a simple recursive function

## 3 Notes

Please read the information at:

<http://www.cs.uiuc.edu/class/sp08/cs421/faq.html>

<http://www.cs.uiuc.edu/class/sp08/cs421/mps/>

The former page tells you how to log on to the EWS systems and hand in your assignment. The latter page explains how to use the automatic grader package.

If you do not yet have an email account, please email Chris Osborn (a TA) at [cosborn3@uiuc.edu](mailto:cosborn3@uiuc.edu).

## 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 the functions will have to conform to any type information supplied, and to yield the same results as any sample executions given.

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

2. (2 pts) Write a function `add_x` that adds the value of `x` to its argument.

```
# let add_x z = ...
val add_x : int -> int = <fun>
# add_x 22;;
- : int = 39
```

3. (4 pts) Write a function `smallest` that takes two integer arguments and returns the integer that is the smallest, if the smallest is bigger than 0, and returns 0 otherwise.

```
# let smallest n m = ...
val smallest : int -> int -> int = <fun>
# smallest 5 3;;
- : int = 3
```

4. (3 pts) Write a function `salutations` that takes a string, which is assumed to be a person's name, and prints out a greeting as follows: If the name is "Sam", it prints out the string

```
"Hi theya, Sam, wachya up ta?"
```

and for any other string, it first prints out "Dear " followed by the given name, followed by ", Welcome to CS421.". In each case, there should be exactly one "newlines" printed in your result, at the end.

```
# let salutations name = ...
val salutations : string -> unit = <fun>
# salutations "Robert";;
Dear Robert, Welcome to CS421.
- : unit = ()
```

5. (6 pts) Write a function `binom n m` that returns the binomial coefficient "n choose m". Your solution should obey the following identities:

$$\binom{n}{m} = (n - m + 1) * \binom{n}{m-1} / m \quad \text{if } m > 0$$
$$\binom{n}{m} = 1 \quad \text{otherwise}$$

Do not worry about handling cases where  $m < 0$  or  $n < 0$ .

```
let rec binom n m = ...
val binom : int -> int -> int = <fun>
# binom 4 2;;
- : int = 6
```