CMPU 101 § 55 · Computer Science I · Spring 2019

Assignment 2

Due February 20, 9:00 a.m.

Problem 1

Write a definition for the recursive function `print-some`, whose behavior is described by this contract:

```
;; PRINT-SOME
;; ----------
;; INPUTS: NUM, a non-negative integer.
;;         STUFF, a number, symbol, or string.
;; OUTPUTS: None
;; SIDE-EFFECT: Prints STUFF, NUM times. At the end of the repeated
;;   output, it prints a newline.
```

Here are some examples of `print-some` being used:

```
> (print-some 3 5)
555
> (print-some 4 '---)
------------
> (print-some -1 "banana")
> (print-some 5 "banana! ")
banana! banana! banana! banana! banana!
```

For the following problems, you should write the contract yourself based on the description provided. Be sure to include sufficient test cases to satisfy yourself that your code works correctly. You can try the examples shown or write your own.

Problem 2

Write a recursive procedure called `zip` that takes two lists `lst1` and `lst2` of equal length as arguments. The procedure `zip` returns a new list that results from combining corresponding members of `lst1` and `lst2` as shown below.

```
(zip '(foo) '(bar))      ⇒  ((foo bar))
(zip '(a b c) '(1 2 3))  ⇒  ((a 1) (b 2) (c 3))
(zip '() '())            ⇒  ()
```

Problem 3

Write a recursive function called `remove-adjacent-duplicates`. This function takes a list `lst` of symbols or numbers as its argument. It finds places in `lst` where a symbol or number is repeated two or more times in a row. It removes the sequence of multiple occurrences and replaces it with a single occurrence of the item.

```
(remove-adjacent-duplicates '(a b b k j j j x 7 7 2))  ⇒  (a b k j x 7 2)
(remove-adjacent-duplicates '(a a b b c c b b a a))    ⇒  (a b c b a)
```

Problem 4

Write a recursive procedure called `count-occurrences` that takes two arguments: a symbol `sym` and a list of symbols `lst`. The procedure should return an integer indicating the number of times that `sym` occurs in `lst`.

```
(count-occurrences 'z '(a z b))      ⇒  1
(count-occurrences 'z '(a z b z c))  ⇒  2
(count-occurrences 'z '(a b c))      ⇒  0
(count-occurrences 'z '())           ⇒  0
```

Submitting

When you are done, save your definitions file, run your code, and save the interactions file as plain text. Then submit them electronically:

```
submit101 g-asmt02 asmt02
```

(where `g-asmt02` is the dropbox you're submitting your work to and `asmt02` is the name of the directory you've saved your work in; change it if you used a different name).