CSC 120 (R Section), Spring 2015 — Quiz #1 Answers

No books, notes, or calculators are allowed. You have 25 minutes to write this quiz. The four questions are worth equal amounts.

Question 1: Consider a function called `mystery1` defined as follows:

```
mystery1 <- function (abc) abc[2] + abc[length(abc)] * abc[length(abc)-1]
```

Write down the value that this function will compute for each of the function calls below:

a) `mystery1 (c(5,3,1,4,6))`
$3 + (6 * 4) \ = \ 27$

b) `mystery1 (c(0.1,5))`
$5 + (5 * 0.1) \ = \ 5.5$

c) `mystery1 (3:10)`
$4 + (10 * 9) \ = \ 94$

d) `mystery1 ((1:10)+1)`
$3 + (11 * 10) \ = \ 113$

Question 2: Consider a function called `mystery2` defined as follows:

```
mystery2 <- function (x,y) {
    s <- 0
    for (i in 1:length(x)) {
        if (x[i] > y)
            s <- s + x[i]
        else
            s <- s + 2*x[i]
    }
    s
}
```

Write down what value this function will compute if it is called as follows:

```
mystery2 (c(3,11,12,0,2), 10)
```

Also write down the new value that the variable `s` has after each of the times that it is assigned a value.

Values assigned to `s`*: 0, 6, 17, 29, 29, 33*

The value returned by the function is 33.

Question 3: Write down a definition for a function called `mul_or_div_by_2` that takes one argument, called `x`, that we assume will be a single number, and returns as its value either `x` divided by 2, if `x` is negative, or `x` times 2, if `x` is positive or zero. For example, `mul_or_div_by_2(3)` should be `6`, and `mul_or_div_by_2(-4)` should be `-2`.

One answer:

```
mul_or_div_by_2 <- function (x) if (x<0) x/2 else x*2
```

Another correct answer, though it's unnecessarily complicated:

```
mul_or_div_by_2 <- function (x) {
    if (x<0)
        r <- x/2
    else
        r <- x*2
    r
}
```

Question 4: Write down a definition for a function called `change_neg_elements` that takes two arguments, called `vec` and `negval`. The first argument, `vec`, will be a numeric vector, and the second, `negval`, will be a single number. The function should return as its value a numeric vector that is like `vec` except that all negative elements are changed to `negval`. Here is the output from an example call of this function:

```
> change_neg_elements (c(3,-1,0,-2,5), 7)
[1] 3 7 0 7 5
```

One answer:

```
change_neg_elements <- function (vec, negval) {
    for (i in 1:length(vec))
        if (vec[i] < 0) vec[i] <- negval
    vec
}
```