Spring 2019 CSC 419 - Programming Languages - Assignment 4

Due date: Thursday, March 21, 2019, before midnight, upload in home folder of class

Please do not copy solutions from the web. Points will be deducted for copied solutions.

a) Compare strings in Python, Ruby, C char definition, and Java String class. Compare these in terms of whether they are primitive to the language or not, string length options, and reliability (just discuss; no need to write a program).
[4 points]

b) Regarding string mutability: what happens in Python and in Ruby when defining a string and changing one of its values? Try running this in Python and Ruby, and write down the code and the result. Also, does this have implications in terms of reliability?
Python should be available on the lab accounts (type python from the terminal). Ruby should also be available from your lab accounts (type irb from the terminal). For Python and Ruby, if you don't have it installed on your computer, you can also run it in your browser via: http://jupyter.org or https://colab.research.google.com
[4 points]

c) Compare categories of arrays (in terms of memory management) for a C++ array declared in a function, a C++ array using the operator **new**, and an array in Ruby and Python (just discuss; no need to program).
[3 points]

d) Compare C++, Ruby and Python in terms of what happens when attempting to access an array element out of range. Try running these and report the code and result.
[3 points]

e) Discuss how Swift enum differs from that of C. Also, write a short program with Swift defining and printing string enums. You can read about enum in SWIFT here: https://developer.apple.com/library/prerelease/content/documentation/Swift/Conceptual/Swift_Programming_Language/Enumerations.html
You can test your program here: https://iswift.org/playground
[4 points]

f) What is side effects in programming languages?
[3 points]

g) Write code in Python that demonstrates two different ways to get side effects in a function (i.e., write two functions, each showing a different way of getting a side effect). Consider ways to get side effects in terms of the scope of the variable and issues of mutability. Call the functions and run the code, showing the side effects.
[10 points]

h) Let the function fun be defined as:

```
int fun(int *k) {
   *k += 4;
   return 3 * (*k) - 1;
```

```
}
```

Suppose fun is used in a program as follows:

```
int main() {
   int i = 10, j = 10, sum1, sum2;
   sum1 = (i / 2) + fun (&i);
   sum2 = fun(&j) + (j / 2);
   return 0;
}
```

What are the values of sum1 and sum2

1) if the operands in the expressions are evaluated left to right?
2) if the operands in the expressions are evaluated right to left?

[6 points]

i) Run the code given above on some system that supports C to determine the values of sum1 and sum2. Explain the results. Take into account the original design of order of evaluation in the C language (versus, for instance, the design of order evaluation in Java).

[3 points; +2 points extra credit: Can you find an older compiler for C that has a different answer? Even if you can't, note what compilers you tried.]