CMSC 330, Practice Problems 1 (SOLUTIONS)

1. Programming languages
   a. Explain how goals for programming languages have changed since the 1960's.
      Shifted from efficiency to ease-of-programming
   b. List 2 desirable attributes for a programming language where Ruby is better than C. Explain why.
      Naturalness of application – Text processing is easier in Ruby
      Cost of use – Small Ruby programs are simpler/quicker to write
   c. List 2 methods for executing a program. Which method is used by Ruby?
      Interpretation & compilation. Ruby is interpreted.

2. Ruby basics
   a. Write a Ruby method foo that takes an integer as a parameter. Call foo with 2 as its argument. Circle & label the formal and actual parameters in your code.
      def foo(x) … end ; foo(2) ; // x = formal param, 2 = actual parameter
   b. Using different Ruby control statements, write 4 code fragments that iterate from i=1 to i=10.
      ```
      1.upto(10) {|i| puts i; }
      (1..10).each {|i| puts i; }
      for i in (1..10) do puts i; end
      i=1; while i<=10 do puts i; i+=1; end
      i=1; do puts i; break if (i+=1)>10 end
      ```
   c. Explain the difference between explicit and implicit variable declarations.
      Explicit – declaration statements declare type of each variable used
      Implicit – first use of a variable declares it and determines its type
   d. List two advantages of static types.
      Helps prevent subtle errors, catches more type errors at compile time
   e. Using Ruby, write a class Teacher that contains an integer field students and an integer field totalStudents that is shared across all objects of class Teacher.
      ```
      class Teacher
          @@totalStudents = 0
          def initialize
              @students = 0
              @@totalStudents += @students
          end
      end
      ```
   f. Give an example of shallow (reference) copy in Ruby.
      x = "a" ; y = x
   g. Give an example of testing for structural equality in Ruby.
      x == y

3. Ruby advanced features
   a. Describe the string matched by the Ruby regular expression /(3{2})/ ?
      $1 = exactly 2 3's, i.e., "33"
   b. Describe the string matched by the Ruby regular expression /([A-Z])/ ?
      $1 = any single uppercase letter
   c. Describe the string matched by the Ruby regular expression /([A-Z]*[0-9])/ ?
      $1 = 0 or more uppercase letters followed by a single digit
   d. Describe the string matched by the Ruby regular expression /(0$)/ ?
      $1 = a 0 at the end of the line
   e. Describe the string matched by the Ruby regular expression /(\.)/ ?
      $1 = a single (literal) period
   f. What is the output of the following Ruby program?

```
"CMSC 330" =~ /([0-9]+)/
puts $1                              // 330
puts $2                              // nil
```

   g. What is the output of the following Ruby program?

```
a = [4,5,6]
a[5] = 7
a.delete_at(1)
puts a                               // 4 6 nil nil 7
a.push(2)
a.push(1)
puts a.pop                           // 1
```

   h. What is the output of the following Ruby program?

```
if  "CMSC 330" =~ /1/ then
        puts "t"
elsif "CMSC 330" !~ /1/ then
        puts "f"                     // f
else
        puts "n"
end
```

   i. What is the output of the following Ruby program?

```
a = ["c", "b", "a"]
puts a                               // c b a
b = a
a.sort!
puts b                               // a b c
```

   j. What is the output of the following Ruby program?

```
a = "CMSC 330 CMSC 351"
b = a.scan(/[A-Z]+/)
puts b                               // CMSC CMSC
a.scan(/[0-9]+ [A-Z]+/) { |x| puts x } // 330 CMSC
```

   k. What is the output of the following Ruby program?

```
a =  {4 => 6, 5 => 7}
puts a[4]                            // 6
puts a[6]                            // nil
puts a.values                    // 6 7 or 7 6
```

l. What is the output of the following Ruby program?

```
h = Hash.new(0)
h["a"] = h["b"]
h["b"] = 7
h["c"] += 2
puts "#{h["a"]} #{h["b"]} #{h["c"]"  // 0 7 2
```

m. What is returned by “file = File.new(filename, "r"); lines = file.readlines( );”?

Array of strings where each string is a line from the file <filename>

n. What is returned by “x = ARGV[0];”?

String for 1st command line parameter

o. Write a Ruby function foo that takes a code block and executes it twice.

```
def foo( ) 2.times{ yield } end
foo( ) { puts “Running” }      // prints “Running Running”
```

4. Ruby programming

a. Write a Ruby program that reads in lines of input from $stdin and remembers all integers (consecutive digits) encountered. Each line of input may contain 0 or more integers or non-integers. The program should stop and print out the list of integers in sorted order (from smallest to largest) when the word “Done!” is encountered.

```
a = Array.new
loop do
  line = $stdin.readline
  break if line =~ /Done\!/
  line.scan(/\d+/) { |x| a.push x.to_i }
end
a.sort!
a.each { |x| puts x }
```

b. Write a Ruby program that reads the name of a text file from the command line, opens the file, reads every line of text in the file, and prints only the lines that contain exclusively the following characters: uppercase and lowercase letters, digits, and underscore. For example, lines that contain space or punctuation should not be printed.

```
// Version that reads entire file into array
file = File.new(ARGV[0], "r")
lines = file.readlines
lines.each{ |line|
  line.chomp!
  if line !~ /[^A-Za-z0-9\_]+/ then
    puts line
  end
}
```

```
// Alternate version that reads file one line at a time
file = File.new(ARGV[0], "r")
until file.eof? do
  line = file.readline
  line.chomp!
  if line !~ /[^A-Za-z0-9\_]+/ then
    puts line
  end
end
```