Problem 1: Basics

Please circle True or False for the following statements:

- Ruby uses a static type system
- True
- False

- Procs and Codeblocks can be used interchangeably
- True
- False

- nil is not an object in Ruby
- True
- False

- In Ruby, types are associated with values
- True
- False

- Ruby has built in support for Regular Expressions
- True
- False

Problem 2: Code Completion

Fill in the blanks of the following code so that it has the desired output

(a) Higher Order Programming

Fill in the blanks so 10 is printed. You cannot hard code blank 1 or blank 2. You must use $x$ and the codeblock.

```ruby
def myfunc(x)
  puts ___BLANK_1____
end
myfunc(3){ ___BLANK_2___ }
```

Blank 1:

```ruby
yield x
```

Blank 2:

```ruby
{|x| x + 7}
```

(b) Creation

Fill in the blanks so that $a$ is a Hash with a default value of an Array of size 3

```ruby
a = ___blank_1___
```

Blank 1:
Hash.new(Array.new(3))

(c) Objects

Fill in the blank so that square has a class variable called length with the value of x

```ruby
class Square
  def initialize(x)
    __Blank_1__
  end
end
```

Blank 1:

```ruby
@@length = x
```

(d) Regex

Fill in the blanks so that "Correct" is printed

```ruby
rxp = /__Blank_1__/n
line1 = "23 years of age"
line2 = "1 year of age"
if rxp =~ line1 && rxp =~ line2
  puts "Correct"
else
  puts "Failed"
```

Blank 1:

```ruby
[0-9]+ years? of age
```

**Problem 3: Coding**

Write a method named procHash(hash). The argument hash is a hash from a numerical key to a Proc. For each key and proc pair, print out "RESULT is the result of Proc(KEY)" where KEY is the key and RESULT is the result of calling the proc associated with the key on said key.

```ruby
Example:
procHash({1=>Proc.new{|x| x * 2}, 2=>Proc.new{|x| x + 3}}) prints out
2 is the result of Proc(1)
5 is the result of Proc(2)
```

```ruby
def procHash(x)
```
for k,v in x
    puts v.call(k).to_s + " is the result of Proc(" + k.to_s + ")"
end
end