HW5 Paige Forsha

Report:

The language I used to complete this assignment was Python. I did this because we use Python frequently at my job. I wanted to practice it more since it has real-world applications for me. Scheme and Python could both complete all 10 problems, but I had to think a bit differently for each language. Firstly and most obviously, they have different syntax. Some of the subroutines/functions had to be renamed because Python cannot use + or - within function names. To remedy this, I tried to keep the names as close as possible and used the standard camel-case naming convention. Secondly, I had to import the math package for the hypotenuse problem. Sin and cos are given to you from the get-go in Scheme. Additionally, I could have imported the NumPy package on the dot product problem and simplified it, but I chose to do it by hand. Scheme has modules, and they are similar to the packages Python has. However, I did not need to use any of them for this set of problems. Another short thing worth noting is that Scheme and Python both have the capability to use lambdas. I used them in each assignment.

In the Scheme version of this assignment, I never had to directly assign a variable. Variable assignment isn't common in Scheme, from what we've learned in class. For the Python version of this assignment, I did assign a variable to make it easy for my loop to work in the dot product problem.

Another thing I noticed while completing this assignment is that Scheme's list manipulation seems more straightforward. Python actually has several kinds of collections—lists, tuples, sets, and dictionaries. Figuring out which one you should use can be a task itself. Scheme has dotted pairs, vectors, and lists. That's 3 "groups" of things, but it's more than likely that you'll end up using a list. It just personally feels less complicated.

The last thing I wanted to mention was symbols. Symbols are not a primitive data type in Python, but they are in Scheme. This changed up how I completed problem 9. The first parameter was a string instead.

Code:

```
#Paige Forsha HW5


import math



#problem 1
def yourname():
  return("Paige Forsha")
```

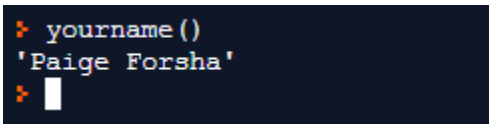
```
#problem 2
def aPlusbx(a, b, x):
  return a + b * x

#problem 3
def hypotenuse(a, b):
  return math.sqrt(a**2 + b**2)

#problem 4
def polarToRect(radius, theta):
    return (radius*math.cos(theta), radius*math.sin(theta))

#problem 5
def dotproduct(list1, list2):
  result = 0
  for i,j in zip(list1,list2):
    result += i*j
  return result

#problem 6
def threshold(min, lst):
  return list(filter(lambda x: x >= min, lst))

#problem 7
def diff(z, lst):
  for n, num in enumerate(lst):
     lst[n] = abs(lst[n] - z)
  return lst

#problem 8
```

```
def buildList(start, end):
  lst = []
  while(start < end):
    lst.append(start)
    start += 1
  return lst


#problem 9
def convert(tag, msg):
  if(tag == "upper"):
    return msg.upper()
  if(tag == "lower"):
    return msg.lower()
  if(tag == "reverse"):
    if isinstance(msg, str):
      return msg [::-1]
    else: #assumes it's a list from now on
      msg.reverse()
      return msg
  if(tag == "square"): #assumes msg is a number
    return msg**2
  else:
    return "not implemented"


#problem 10
#yes i realize there was probably an easier way but i just brute
forced it
def stripSpaces(string):
  string = string.replace(" ","")
  string = string.replace("\n","")
  string = string.replace("\t","")
```

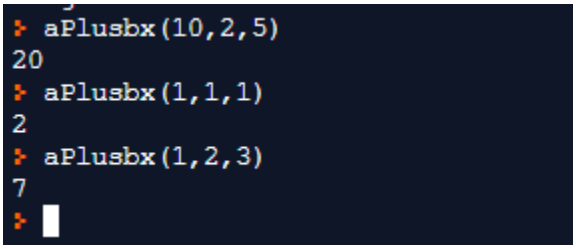
```
  string = string.replace("\f","")
  string = string.replace("\v","")
  string = string.replace("\r","")
  return string
```

Screenshots:

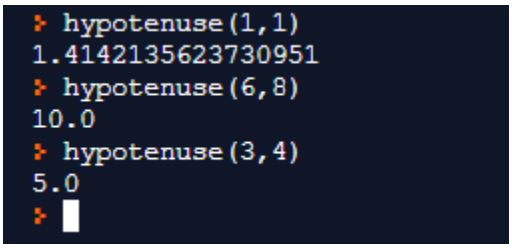
Problem 1



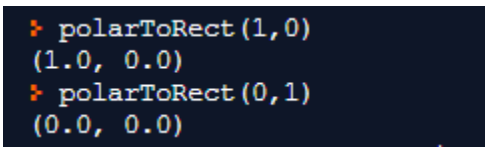
Problem 2



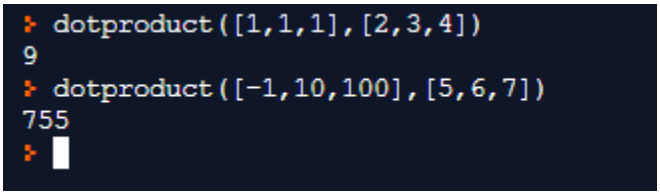
Problem 3



Problem 4



Problem 5



Problem 6

```
> threshold(5,[1,2,3,4,5,6,7,8,9])
[5, 6, 7, 8, 9]
> threshold(5,[2,5,4,7,8,6,3])
[5, 7, 8, 6]
>
```

Problem 7

```
> diff(10,[8,9,10,11,12])
[2, 1, 0, 1, 2]
> diff(10,[11,9,100,3,-4,10])
[1, 1, 90, 7, 14, 0]
>
```

Problem 8

```
> buildList(0,5)
[0, 1, 2, 3, 4]
> buildList(10,20)
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
> buildList(5,0)
[]
>
```

Problem 9

```
> convert("upper", "this is a test")
'THIS IS A TEST'
> convert("lower", "THIS is A TEST")
'this is a test'
> convert("reverse", "THIS is A TEST")
'TSET A si SIHT'
> convert("reverse", [1,2,3,4,5])
[5, 4, 3, 2, 1]
> convert("square", 2)
4
> convert("square", 5)
25
> convert("sdfsdf", 5)
'not implemented'
>
```

Problem 10

```
> stripSpaces("   skfskdj  s s s s s ")
'skfskdjsssss'
> stripSpaces("apples are a good fruit ")
'applesareagoodfruit'
>
```