CSE 101: Introduction to Computational and Algorithmic Thinking

Exam 1 Study Questions

Programming Questions: Short Answer

1. Give the output of each of the following short Python programs. Note that there are no syntax errors in any of this code.

a.
```
a = 14 // 3
print(str(a))
```

a. 4

b.
```
b = 20
b += 7
print(str(b))
```

b. 27

c.
```
c = 5
c = c**3
print(str(c))
```

c. 125

d.
```
a = 13 / 2
print(a)
```

d. 6.5

e.
```
b = 29 % 9
print(b)
```

e. 2

f.
```
c = 19 // 3
print(c)
```

f. 6

g.
```
d = 2 ** 5
print(d)
```

g. 32

h.
```
e = 'WolfieSeawolf'
print(e[5])
```

h. e

i.
```
f = 'Hi' * 3
print(f)
```

i. HiHiHi

j.
```
d = 9
d = 8 + d - 2 * d
print(str(d))
```

j. -1

k.
```
e = 6 / 4
print(str(e))
```

k. 1.5

l.
```
t = 7
t *= 2 + 1  # not a typo
print(t)
```

l. 21

m.
```
d = 5
e = 8
f = 4
if e - f == f - 1:
    print('X')
elif d - f > e:
    print('Y')
else:
    print('Z')
```

m. Z

n.
```
p = 2
q = 7
r = 6
if p ** 0 > 1:
    print('A')
elif q - r == q % r:
   print('B')
else:
    print('C')
```

n. B

o.
```
a = ["ABCDEF", "GHIJK", "MN"]
print(str(len(a)))
```

o. 3

p.
```
b = ["ABCDEF", "GHIJK", "MN"]
print(str(len(b[1])))
```

p. 5

q.
```
letters = "ZYXWVUTSR"
d = letters[len(letters) % 5]
print(d)
```

q. V

r.
```
name = "ADALOVELACE"
e = ""
for r in name:
    if r != "A":
        e += r
print(e)
```

r. DLOVELCE

s.
```
total = 0
while total < 10:
    for i in range(3):
        total += i
    total *= 2
    print(total)
```

s. 6 18

t.
```
g = [[1,3,7], [2,6,8,4], [9], [5,2]]
print(len(g[2]))
```

t. 1

u.
```
h = [[1,3,7], [2,6,8,4], [9], [5,2]]
print(h[1])
```

u. [2,6,8,4]

v.
```
r = ['stony','brook','university']
print(r[1])
```

v. 'brook'

w.
```
GPA = 3.6
if GPA > 3:
    if GPA < 3.5:
        print('1')
    elif GPA > 3.8:
        print('2')
    else:
        print('3')
elif GPA > 2:
    print('4')
else:
    print('5')
```

w. 3

x.
```
month = 10
if month > 2:
    if month < 6:
        print('A')
    else:
        print('B')
if month > 9:
    if month < 12:
        print('D')
    else:
        print('E')
```

x. B D

y.
```
r = 5
if r > 4:
    if r < 6:
        print('A')
    else:
        print('B')
if r < 3:
    if r > 1:
        print('D')
else:
    print('F')
```

y. A F

2. Give the output of each of the following short Python programs. Note that there are no syntax errors in any of this code.

a.
```
for i in range(4):
    print(i)
```

a. 0 1 2 3

b.
```
count = 0
for i in range(10, 18):
    if i % 3 == 1:
        count += 1
print(count)
```

b. 3

c.
```
nums = [5, 8, 2, 4]
for i in nums:
    print(i+1)
```

c. 6 9 3 5

d.
```
nums = [1, 3, 5, 7, 9, 2, 4, 6, 8]
for i in range(4):
        print(nums[i+1])
```

d. 3 5 7 9

e.
```
nums = [2, 7, 3, 1, 4, 5, 8]
for i in nums:
    if i % 2 == 0:
        print(i)
```

e. 2 4 8

f.
```
n = 7
while n >= 0:
    if n % 3 == 0:
        print(n)
    n -= 1
```

f. 6 3 0

g.
```
for i in range(1, 8, 2):
    print(i)
```

g. 1 3 5 7

3. Suppose we have the following line of code:

```
groups = [[8, 6, 7, 4], [2, 9, 11], [3, 15, 5, 14, 29], [17, 19]]
```

Give the output for each of the following code fragments. Write "Error" if the code would cause a program crash.

a. `print(len(groups))`

a. 4

b. `print(len(groups[1]))`

b. 3

c. `print(groups[2])`

c. [3, 15, 5, 14, 29]

d. `print(groups[1][2])`

d. 11

e. `print(groups[3][2])`

e. Error

4. What value will be stored in the variable `total` after the following code has been executed?

```
total = 0
for i in range(5):
    total += i
```

4. 10

5. What value will be stored in the variable `total` after the following code has been executed?

```
total = 0
for i in range(4):
    total += i
    total += total
```

5. 22

6. What value will be stored in the variable `result` after the following code has been executed?

```
words = ['car', 'truck', 'boat', 'plane', 'bike']
result = ''
for i in range(len(words)-1):
    result += words[i] + '--'
```

6. `'car--truck--boat--plane--'`

7. What value will be stored in the variable `result` after the following code has been executed?

```
nums = [6, 2, 3, 9, 8, 10, 5, 12, 7, 0, 22]
result = []
for n in nums:
    if n % 2 == 0:
        result.append(n)
```

7. `[6, 2, 8, 10, 12, 0, 22]`

8. What value will be stored in the variable `result` after the following code has been executed?

```
words = ['car', 'truck', 'boat', 'plane', 'bike']
result = []
for i in range(len(words)-1):
    result += [words[i]]
result
```

8. `['car', 'truck', 'boat', 'plane']`

9. Fill in the blank to complete the Python code given below. Write only one line of code for each part.

   a. The incomplete code below would compute the value of x in the following way: if a is divisible by b, then $x = 20 - \frac{3a+7}{b+2}$, otherwise $x = 20$. Assume that a and b (with $b \neq 0, b \neq -2$) are stored in variables of the same names. Use floating-point division.

```
if a % b == 0:

    _______________________________________________
else:
    x = 20
```

```
x = 20 - (3*a+7)/(b+2)
```

b. The incomplete code below would **overwrite** all the odd values in a list of integers called `nums` with the value -1. For instance, if `nums` were `[1, 4, 2, 7, 11, 8, 6]`, then `nums` would change to `[-1, 4, 2, -1, -1, 8, 6]`. Complete the code by writing a **while-loop**.

```
i = 0

_______________________________________________
    if nums[i] % 2 == 1:
        nums[i] = -1
    i += 1
```

```
while i < len(nums):
```

c. The incomplete code below would take a string (all **lowercase**) called `s` and **print** the vowels in the order they appear in the string. For instance, if `s` were `'qwertyuioasd'`, the code would print
`e u i o a`

```
for ch in s:

    _______________________________________________
        print(ch)
```

```
if ch in 'aeiou'
```

d. The incomplete function `count_ages` below would take a list of ages called `ages` and **return** the count of how many of the values are above 50. For instance, if `ages` were `[27, 16, 56, 48, 87]`, the function would return 2.

```
def count_ages (ages):
    count = 1
    for a in ages:
        if a > 50:
            count += 1

    _______________________________________________
```

```
return count-1
```

10. Each of the Python codes below contains a bug. Write the **number** of line containing the bug and then fix it. You may only change **one** line.

a. The function `total(n)` takes an integer parameter and returns the sum of values from 0 through `n-1`.

```
1. def total(n):
2.     s = 0
```

```
3.      for i in range(n):
4.          s += i
5.          return s
```

Line 5 contains a bug.

It should be not indented .

b. The function `compute(x, y)` below takes two numbers as parameters, returning the result in the following way (assume the arguments are always valid with $y \neq 0$):

- if `x > y`, return $3 + \frac{2(x+6)}{y}$
- if `x <= y`, return $\frac{(x+6)}{y}$

```
1. def compute(x,y):
2.     if x > y:
3.         result = (3 + 2*(x+6))/y
4.     else:
5.         result = (x+6)/y
6.     return result
```

Line 3 contains a bug.

It should be `result = 3 + 2*(x+6)/y` .

c. The function `compute(x, y)` below takes two numbers as parameters, returning the result in the following way (assume the arguments are always valid with $x \neq 0, y \neq 0$):

- if `y > 0`, return $\frac{4(x+7)}{y}$
- if `y <= 0`, return $\frac{(y+4)}{x}$

```
1. def compute(x,y):
2.     if y > 0:
3.         result = 4(x+7)/y
4.     else:
5.         result = (y+4)/x
6.     return result
```

Line 3 contains a bug.

It should be `result = 4*(x+7)/y` .

d. The function `pay_value(amount, price)` below takes two parameters in this order: the number of products that a person buys, called `amount`, and the total price for all products, called `price`. The function should compute an amount-based discount and return the final price. The rules for calculating the amount-based discount are:

- if the amount is less than 3, the person receives a 5% discount off the price
- if the amount is between 3 and 8 (inclusive), the person receives a 15% discount off the price

- if the amount is between 9 and 10 (inclusive), the person receives a 25% discount off the price
- if the amount is at least 11, the person receives a 35% discount off the price

```
1.  def pay_value(amount, price):
2.      if amount < 3:
3.          result = price * (1 - 0.05)
4.      elif amount >= 11:
5.          result = price * (1 - 0.35)
6.      elif amount < 9:
7.          result =  price * (1 - 0.25)
8.      else:
9.         result =  price * (1 - 0.15)
10.     return result
```

Line 6 contains a bug.

It should be `elif amount >= 9:` .

e. The function `count_days(month,day)` below takes two parameters in this order: `month` (from 1 to 12), and `day` (from 1 to 31). The function should compute and return how many days have passed through that day of the year (including that day). For instance, if $month = 3$ and $day = 12$, then 71 ($71 = 31 + 28 + 12$) should be returned. Assume there are 28 days in February.

```
1.def count_days(month, day):
2.    count = 0
3.    mm = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
4.    for a in range(month):
5.        count += mm[a]
6.    count += day
7.    return count
```

Line 4 contains a bug.

It should be `for a in range(month-1)` .

11. The Python code below contains a bug. Write the **number** of line containing the bug and then fix it. You may only change **one** line.

    The function `purchase(prices, max_total)` below takes two parameters in this order: `prices` (a list of integers) and `max_total` (an integer). Starting with the leftmost element in `prices`, the function should add values from `prices` while `total` is at most `max_total`. For example, if `prices = [3, 2, 1, 5, 4, 6]` and `max_total = 7`, the function returns 6 because $3+2+1=6$ and $3+2+1+5=11$, which is greater than 7. Another example: if `prices = [3, 2, 1, 5, 4, 6]` and `max_total = 25`, the function returns 21 because $3+2+1+5+4+6=21$.

```
1. def purchase(prices, max_total):
2.     i = 0
3.     total = 0
4.     while total < max_total and i < len(prices):
5.         if prices[i] + total <= max_total:
6.             total += prices[i]
7.         i -= 1
8.     return total
```

    Line 7 contains a bug.

    It should be `i += 1`.

12. The Python code below is supposed to take a list of strings called `strs` and print the count of how many of the strings contains character k and r, respectively. For instance, if `strs` were `['Stony', 'Brook', 'University', 'Computer', 'Science']`, then the output should be `1 3`. Fix the code by identifying the one line of code with the error and indicating your correction below.

```
1.  k = 1
2.  r = 1
3.  i = len(strs)
4.  while i > 0:
5.      if 'k' in strs[i-1]:
6.          k += 1
7.      if 'r' in strs[i-1]:
8.          r += 1
9.      i += 1
10. print(k-1)
11. print(r-1)
```

    Line 9 contains a bug.

    It should be `i -= 1`.

Programming Questions

13. Write a Python function `retail(wholesale)` that returns the cost of a sofa based on the following scenario. At Lou's Discount Furniture Store, Lou marks the furniture at twice his wholesale cost, plus $80. He then marks the price down by 15% to determine the retail price. The function takes the wholesale cost as an argument and returns the retail price.

```
def retail(wholesale):
    price = 2*wholesale + 80
    price *= 0.85
    return price
```

14. Write a Python function `total_seconds(hours, minutes, seconds)` that takes arguments representing a time duration in hours, minutes and seconds, and then returns the equivalent total number of seconds. For example, if the three arguments were 3, 28 and 42, in that order, the function would return 12522 because 3 hours, 28 minutes and 42 seconds are equivalent to 12,522 seconds.

```
def total_seconds(hours, minutes, seconds):
    return hours*60*60 + minutes*60 + seconds
```

15. Write a Python function `wind_chill(temp_f, velocity_mph)` that calculates and returns the wind chill factor (W), given the air temperature in Fahrenheit (`temp_f`) and wind velocity in miles per hour (`velocity_mph`). The wind chill formula is given as follows, where T is the air temperature in Fahrenheit and V is the wind velocity in miles per hour:

$$W = 35.74 + 0.6215T - 35.75V^{0.16} + 0.4275TV^{0.16}$$

For example, if `temp_f` is 30.0 and `velocity_mph` is 20.0, the function should return approximately 17.361783756466327.

```
def wind_chill(temp_f, velocity_mph):
    return 35.74 + 0.6215*temp_f - 35.75*velocity_mph**0.16 +
           0.4275*temp_f*velocity_mph**0.16
```

16. Write a Python function `payday(hourly_wage, hours_worked)` that calculates and returns how much a person is paid for the week. The first argument gives the hourly wage, and the second argument gives number of hours worked that week. The employee is paid according to the following scheme:

- for each hour worked up to and including the 40th hour, the employee earns his normal hourly wage
- for each hour between 41 and 50 hours, the employee earns 1.5 times his hourly wage
- for each hour worked over 50 hours, the employee earns 2.0 times his hourly wage

The function calculates and returns the total pay for the person. For instance, for 54 hours worked at a wage of $8.25 per hour, the function would compute the total pay as $519.75.

```
def payday(hourly_wage, hours_worked):
    pay = 0
    if hours_worked <= 40:
        pay = hourly_wage * hours_worked
    elif hours_worked <= 50:
        pay = hourly_wage * 40 + 1.5 * hourly_wage * (hours_worked-40)
    else:
        pay = hourly_wage * 40 + 1.5 * hourly_wage * 10 + \
              (hours_worked-50) * 2 * hourly_wage
    return pay
```

17. Write a Python function `multiples_of_3(low, high)` that returns a list containing all of the values from `low` through `high` (inclusive) that are evenly divisible by 3, and only those values. For example, if `low` is 5 and `high` is 21, the function returns the list `[6, 9, 12, 15, 18, 21]`.

```
def multiples_of_3(low, high):
    result = []
    for n in range(low, high+1):
        if n % 3 == 0:
            result += [n]
    return result
```

18. Write a Python function `shortest(names)` that takes a list of strings containing all the first names of undergraduate students at Stony Brook and returns the shortest name in the list that is at least five characters long. If two or more names have the same shortest length, then the function may return any one of them. Assume that there is at least one name in the list that has at least five characters. Also assume that no name in the list has more than 20 characters in it. Your function should work for a list that could have hundreds or even thousands of names in it.

    For example, if the `names` argument were `['Fred', 'Pauline', 'Jennifer', 'Tom']`, the function would return `'Pauline'` because that is the shortest name that has at least five characters in it.

```
def shortest(names):
    shortIndex = 0
    shortLen = 20
    for i in range(len(names)):
        if len(names[i]) < shortLen and len(names[i]) >= 5:
            shortIndex = i
            shortLen = len(names[i])
    return names[shortIndex]
```

19. Write a Python function `one_copy(nums)` that takes a list of integers as its argument and returns a list containing exactly one instance of each number appearing in the argument. In other words, the duplicated values are eliminated.

    For example, if the list `nums` were `[8, 6, 7, 4, 2, 6, 9, 8, 7, 3, 4, 2]`, the returned list would contain only the seven values 8, 6, 7, 4, 2, 9 and 3 (in any order) with no duplicated values.

```
def one_copy(nums):
    result = []
    for n in nums:
        if n not in result:
            result.append(n)
    return result
```

Computer Hardware and Software Basics

No answers are provided for these questions, but you should be able to answer them by reviewing the lecture notes and/or reading the relevant chapters of the textbook.

20. What is **Moore's law**? What physical limitations may prevent it from holding true in the future?

21. If Moore's Law holds over the next decade, about how much faster can we expect computers to be in 10 years? Show work or briefly explain your answer.

22. What is the connection between **algorithms** and computer programming?

23. Explain the major differences between **machine language** and **high-level programming languages** like Python.

24. Briefly describe the **stored-program concept**. Why is it important?

25. Briefly explain what occurs during a computer's **fetch-decode-execute cycle**.

26. Briefly describe the major components of the **von Neumann architecture** and how they cooperate to form a functioning computer.

27. Briefly describe the major components of a **central processing unit**.

Fundamental Concepts in Programming

No answers are provided for these questions, but you should be able to answer them by reviewing the lecture notes and/or reading the relevant chapters of the textbook.

28. What is a **floating-point number**?

29. What is the relationship between the following concepts: machine language, the Python programming language and the Python interpreter?

30. In programming, when do you think it makes sense to declare a new **variable**?

31. What does it mean to say that Python is a **case-sensitive language**?

32. How can we write **self-documenting code**? What would code that is *not* self-documenting look like?

33. How are functions in programming similar to functions in mathematics? How are they different?