Midterm Practice Exam CS2021 - Spring 2017
Name_____ ID#_____

Next to each code fragment below, write what is printed when the code fragment is executed in Python. If the code would cause an error, write ERROR and give a brief explanation.

1. What Would Python Print?

```
word = 'Python!'
print(word[1])
print(word[-1])
print(word[6])
```

2. What Would Python Print?

```
sad = False
num = 0
print(not sad and 1/num == 1)
```

3. What Would Python Print?

```
list1 = [1, 2, 3]
list2 = [4] + list1
print(list1)
print(list2)
```

4. What Would Python Print?

```
list1 = [1, 2, 3, 4]
list2 = list1.append('101')
print(list1)
print(list2)
```

5. What Would Python Print?

```
a = 'cat+dog=friends'
print(a[0:4])
print(a[:3] + a[8:])
```

6. What Would Python Print?

```
def full(n):
  total, stop = 1, 1
  while n > stop:
    total, n = total + n, n-1
  return total

print(full(6))
```

7. What Would Python Print?

```
def exp_decay(n):
  if n % 2 != 0:
    return
  while n > 0:
    print(n)
    n = n // 2

exp_decay(48)
```

8. What Would Python Print?

```
a = ['dog']
b = a
a[0] = 'cat'
print(a)
print(b)
```

9. What Would Python Print?

```
a = ['dog', 'cat', 'bird' ]
b = a[1:] + [a[0]]
print(a)
print(b)
```

10. What Would Python Print?

```
a = ['dog', 'cat', 'bird' ]
b = a
b = b[1:] + [b[0]]
print(a)
print(b)
```

11. Fix the errors in the following function so all factors of n are printed?

```
def factors(n):
    x = n
    while x > 0:
    if n % x == 0:
    print(x)
    x -= 1

>>>factors(15)
15 5 3 1
```

12. Python has four main collection types:

(a)strings (b)lists (c)dictionaries (d) tuples

i) Which are iterable?

ii) Which are sequences?

iii) Which are mutable?

iv) Which are immutable?

13. What Would Python Print?

```
def mystery(lst):
  res = []
  for x in lst:
     res = [x] + res
  return res

print(mystery([1, 2 ,3, 4]))
```

14. What Would Python Print?

```
f = lambda st:
          st[:len(st)//2]
print(f([1,2,3,4,5]))
```

15. What Would Python Print?

```
def lucky():
  def charms():
    print('citrus')
  print('drink')
  return charms

lucky()()
```

Questions 18-32. What Would Python Print?

In each of the following, provide an exact representation of what python would print using interactive commands.

18.

>>> yo = [1]

>>> len(yo)

_____

19.

>>> 1 in yo

_____

20.

>>> yo + [2, 3]

_____

21.

>>> [0] + yo

_____

22.

>>> yo * 3

_____

23.

>>> zoo = [[1, 2], [3, 4, 5]]

>>> len(zoo)

_____

24.

```
>>> list(map(lambda x: x*x, [1, 2, 3, 4]))
```

_____

25.

```
>>> list(filter(lambda x: x % 2 == 0, [1, 2, 3, 4]))
```

_____

26.

```
>>> seq = [1, 2, 3, 4]
>>> [x * x for x in seq if x % 2 == 0]
```

27.

```
>>> [x*x for x in range(5)]
```

_____

28.

```
>>> [n for n in range(10) if n % 2 == 0]
```

_____

29.

```
>>> ones = [1 for i in ["hi", "bye", "you"]]
>>> ones + [str(i) for i in [1, 2, 3]]
```

_____

30.

```
>>> [i+5 for i in [n for n in range(1,4)]]
```

_____

31.

```
>>> type(2)
```

_____

32.

```
>>> type(1.5)
```

_____

33.

```
>>> xo = [1, 3, [5, 7], 9]
>>> _________________
```

7

34.

```
>>> xo = [[7]]
>>> _________________
```

7

35.

```
>>> xo = [1, [2, [3, [4, [5, [6, [7]]]]]]]
>>> _________________
```

7

.

36.

```
>>> pokemon = {'pikachu': 25, 'dragonair': 148,
'mew': 151}
>>> ____________
```

25

37.

```
>>> pokemon['jolteon'] = pokemon[‘mew’]
>>> _______
```

151

Part II.

1. **Write a Python function if_this_not_that :** Define a function which takes a list of integers *i_list*, and an integer *this*, and for each element in *i_list* if the element is larger than *this* then print the element, otherwise prints string 'that'.

```
def if_this_not_that(i_list, this):
    """
    >>> original_list = [1, 2, 3, 4, 5]
    >>> if_this_not_that(original_list, 3)
    that
    that
    that
    4
    5
    """
```

2. **Write a Python function replace_all :** Define a function that given a dictionary d, replaces all occurrences of x as a value (not a key) with y.

```
def replace_all(d, x, y):
    """
    >>> d = {'foo': 2, 'bar': 3, 'garply': 3, 'xyzzy': 99}
    >>> replace_all(d, 3, 'poof')
    >>> d == {'foo': 2, 'bar': 'poof', 'garply': 'poof', 'xyzzy': 99}
    True
    ""
```

3.. **Write Python functions (please write answers on back of page).** While planning the potluck, the staff decided to try and guess the number of people that would show up. In order to do this, they decided to define a new abstract data type to record everyone's predictions. Of course, the staff is bad at computer science, so they need your help to make this work!

We want to make a prediction abstract data type that will record both a person's name as well as their guess for the number of attendees. Based on the provided constructor make_prediction, fill in the definitions for the get_name and get_guess selectors.

```
def make_prediction(name, guess):
    return [name, guess]
def get_name(prediction):
 """Gets the name of the person who made the given prediction.
>>> get_name(make_prediction('eric', 25))
'eric'
"""
def get_guess(prediction):
"""Gets the number of attendees that this prediction expected to
show up to the potluck.
>>> get_name(make_prediction('eric', 25))
 25
"""
```

4. **Write Python functions.** Now complete the print_winner function. It takes a sequence of predictions and the actual number of attendees, and prints a congratulatory message based on whose guess was closest. You may assume that the sequence of predictions is non-empty. Ties should go to the person whose prediction appears earliest in the sequence. Remember to respect data abstraction.

```
def print_winner(predictions, correct_num):
 """Given a sequence of predictions (predictions) and the actual
number of attendees (correct_num), print the message '___ is the
winner', where the blank is filled in with the name of the person who
made the winning prediction.

>>> albert_pred = make_prediction('albert', 10000)
>>> brian_pred = make_prediction('brian', 85)
>>> mark_pred = make_prediction('mark', 97)
>>> preds = [albert_pred, brian_pred, mark_pred]
>>> print_winner(preds, 83)
brian is the winner
>>> preds2 = [make_prediction('rohan', 90),
make_prediction('jeffrey', 70)]
>>> print_winner(preds2, 80)
rohan is the winner
 """
```