Faculty of
Information Technology
King Mongkut's Institute of Technology Ladkrabang

Name:...

Student ID:...

Assignment # 2: Stacks

1. In your own words, describe the definition of a stack and its properties.

2. Given a class of Stack which is defined as follows:

```
class ArrayStack:
  def __init__(self):
    """ Create an empty stack. """
    self._data = []    # Initiate a nonpublic list instance

  def __len__(self):
    """ Return the number of elements in the stack. """
    return len(self._data)

  def is_empty(self):
    """ Return True if the stack is empty. """
    return len(self._data) == 0

  def push(self, element):
    """ Add an element to the top of the stack. """
      self._data.append(element)  # new item stored at end of
     list

  def top(self):
     """ Return (but do not remove) the element at the top of
     the stack. Raise an exception if the stack is empty. """
    if self.is_empty():
      print('Stack is empty')
      raise Empty('Stack is empty') # Calling subclass Empty
    return self._data[-1] # the last item in the list

  def pop(self):
      """ Remove and return the element from the top of the
     stack. Raise an exception if the stack is empty. """
    if self.is_empty():
      print('Stack is empty')
        raise Exception('Stack is empty') # Alternate way to
     call subclass Empty
    return self._data.pop() # remove last item from the list
```

A method to create an empty stack is:

S = ArrayStack()

What values are returned during the following series of stack operations, if executed upon an initially empty stack?

Operation	Return Value	Stack values
S.push(7)		
S.push(2)		
S.pop()		
S.push(5)		
S.push(9)		
S.pop()		
S.top()		
S.pop()		
S.push(9)		
len(S)		
S.push(0)		
S.pop()		
S.is_empty()		
S.push(4)		
S.pop()		
S.pop()		
len(S)		

3. Based on a Stack class as defined in question 2, write the pseudocodes or python codes for the following tasks.
    - Create 3 stacks: A, B and C.
    - Add new items including “Ant”, “Bird”, “Cat”, and “Dog”, respectively, into stack “A”.
    - Add new items including “Tony”, “Sara”, and “Adam”, respectively, into stack “B”.

- Transfer the “Dog” and “Cat” items from stack “A” to stack “B”.

- Add “Bear” and “Panda” items into the stack “A”. In the final result, all items in the stack “A” are sorted from “A” to “Z” where “Bear” is in the top of the stack and “Panda” is in the bottom.
  Hint: Use stack C to store elements that are not in the correct order.

- Remove all items in stack “B” by using a loop.

4. Based on a Stack class as defined in question 2, implement a function that removes all items in a stack using Python code or pseudocode.

   Hint: Use a loop or recursive method.

```
class ArrayStack:
        def __init__(self):
                ...
        def __len__(self):
                ...
        def pop_all(self):
                #Add your code here
```

5. Based on Stack class as defined in question 2, implement a function that reverses a list of elements by pushing them onto a stack in one order, and writing them back to the list in reversed order using Python code or pseudocode.

   Hint: Use a loop and 2 empty stacks to store elements similar to the Tower of Hanoi example.
   Example Case: A stack of [1, 2, 3] should be reversed and returned as [3, 2, 1].

```
class ArrayStack:
        def __init__(self):
                …
        def __len__(self):
                …
        def reverse_list(self):
                #Add your code here
```

6. Suppose you have three nonempty stacks X=[1, 2, 3], Y=[4, 5], and Z=[6, 7, 8, 9], describe a sequence of operations using Python or pseudocode that results in X=[1, 2, 3] and Y=[6, 7, 8, 9, 4, 5] with both sets of those elements in their original order.
    - Note: order left-to-right is bottom-to-top.

7. Suppose an initially empty stack S has executed a total of 25 push operations, 12 top operations, and 10 pop operations, 3 of which raised Empty errors that were caught and ignored. What is the current size of S? Also, show your calculation of how you obtain the size.

8. Write a recursive Python or pseudocode function that finds the maximum value in a sequence of *n* elements (can be either ordered or unordered) without using any loops. Function max(a,b) returns a max value between a and b.
   Hint: use the last element as a stop condition for a recursive.

```
def find_max(data,index=0):
    if #Add your code here
        return #Add your code here
    return max(#Add your code here, #Add your code here)
```

9. In Python or pseudocode, write a recursive algorithm to compute the product of two positive integers, m and n, using only addition and subtraction.

   For example, 4 * 3 = 4 + 4 + 4 = 12.

```
def product(m, n):
   if #Add your code here
      return #Add your code here
   else:
      return #Add your code here
```