Disclaimer: This mock exam is only for practice. It was made by tutors in the Polytechnic Tutoring Center and is not representative of the actual exam given by the CS Department.

1 Write a recursive function that takes a list, starting index and ending index as a parameters and prints it in reverse.

Sample Output:

```
ls = [1,2,3,4]
revPrint(ls, 0, 3)
>> 4 3 2 1
```

Code:

```python
def printReverse(s, low, high):
    if low > high:
        return
    printReverse(s, low+1, high)
    print(s[low], end=" ")
```

2 Circle the faster runtime:

- $O(n \log(n))$ or $O(\sqrt{n})$
- $O(\sqrt{n})$ or $O(\log(n))$
- $O(n \log(n))$ or $O(n^{1.25})$
3. Let \( f(x,n) = \sum_{i=1}^{n} \frac{x}{i} = \frac{x}{1} + \frac{x}{2} + \frac{x}{3} \ldots \), so \( f(4,3) = \frac{4}{1} + \frac{4}{2} + \frac{4}{3} = 7.333 \). Write a function for \( f \). State your runtime.

**Code:**

```python
def sumfunc(x, n):
    retval = 0
    for i in range(1, n+1):
        retval += x/i
    return retval
```

4. What is the runtime of the following code snippets?

a. `def function3(lst, low, high):
   if (low >= high):
       return 3
   for elem in lst:
       elem += 2
   return function3(lst, low+1, high-1)`
   \( O(n^2) \)

b. `def function2(lst):
   if (len(lst) == 1):
       lst[0] = 0
       return 2
   return function2(lst[:len(lst)//2])`
   \( O(n) \)

c. `def function1(lst, lst2):
   for elem in lst:
       if (elem in lst2):
           print('iteration')`
   \( O(m*n) \)
5. If \( A = [0,0,0,0,0,0] \), \( B = [3,1,6,2] \), what does \( A \) and \( B \) look like after \text{function2}(B) \) and \text{function3}(A,2,len(A)-1) \)? (Refer to Question 4)

\[
A = [0,0,0,0,0,0] \\
B = [3,1,6,2]
\]

6. Write a generator function that provides the values for a harmonic series of \( n \) elements. Hint: Harmonic series is 1, 1/2, 1/3...

Sample Output:

\[
\text{iters} = 4 \\
\text{display_list} = \text{list(harmonic(iters))} \\
\text{display_list} \\
\text{>> [1.0,0.5,0.33,0.25]}
\]

Code:

```python
def harmonic(n):
    for i in range(1:n+1):
        yield 1/i
```

7. Given a non-empty list with integers, write a function \text{separate_num(lst)} \) to separate a list of even numbers and odd numbers and returns a list that contains all the odd numbers in the front and all even numbers in the back.

Example: an input list \([3,15,44,2,51,89,20]\) to \text{separate_num(lst)} \) will return \([3,15,51,89,44,2,20]\)

Requirement: \( O(n) \) runtime and in place

(1) Do the implementation of \text{separate_num(lst)} \text{iteratively}
Code:

def reset(lst):
    i = 0
    j = len(lst)-1
    while i < j:
        if lst[i] % 2 == 0:
            if lst[j] % 2 == 1:
                temp = lst[i]
                lst[i] = lst[j]
                lst[j] = temp
                i += 1
                j -= 1
            else:
                j -= 1
        else:
            i += 1
    return lst

(2) Do the implementation of separate_num(lst) recursively with a helper function

Code:

def reset(lst):
    return helper(lst, 0, len(lst)-1)

def helper(lst, low, high):
    if low > high:
        return lst
    else:
if lst[low] % 2 == 0:
    if lst[high] % 2 == 1:
        temp = lst[low]
        lst[low] = lst[high]
        lst[high] = temp
        return helper(lst, low+1, high-1)
    else:
        return helper(lst, low, high-1)
else:
    return helper(lst, low+1, high)