CS 5012: Foundations of Computer Science

Asymptotic Complexity Exercise

Given the following code snippets, provide the time complexity in the form of Big-O notation. Justify your response and state any assumptions made. Treat these functions as constant runtime: print(), append()

In class review:

1) Finding the Maximum Element

```
def findMax(arr):
    max_val = arr[0]
    for num in arr:
        if num > max_val:
                max_val = num
    return max_val
```

The asymptotic complexity of this algorithm is: __________________

2) Calculating the Sum of an Array

```
def arraySum(arr):
    sum = 0
    for num in arr:
        sum += num
    return sum
```

The asymptotic complexity of this algorithm is: __________________

3) Recursive Fibonacci Sequence

```
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return(fibonacci(n-1) + fibonacci(n-2))
```

The asymptotic complexity of this algorithm is: __________________

4) Binary Search Algorithm

```
def binarySearch(arr, low, high, x):
    if high >= low:
        mid = (high + low) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binarySearch(arr, low, mid - 1, x)
        else:
            return binarySearch(arr, mid + 1, high, x)
    else:
        return -1
```

The asymptotic complexity of this algorithm is: ________________

5) Insertion Sort Algorithm

```
def insertionSort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i-1
        while j >=0 and key < arr[j] :
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
```

The asymptotic complexity of this algorithm is: ________________

In class group assignment:

6) Calculating the Factorial of a Number (2pts)

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)
```

The asymptotic complexity of this algorithm is: ________________

7) Checking for Duplicates in an Array (2pts)

```
def checkDuplicates(arr):
    for i in range(len(arr)):
        for j in range(i + 1, len(arr)):
            if arr[i] == arr[j]:
                return True
    return False
```

The asymptotic complexity of this algorithm is: _________________

8) Linear Search Algorithm (2pts)

```
def linearSearch(arr, x):
    for i in range(len(arr)):
        if arr[i] == x:
            return i
    return -1
```

The asymptotic complexity of this algorithm is: _________________

9) Merging Two Sorted Arrays (2pts)

```
def mergeArrays(arr1, arr2):
    result = []
    i, j = 0, 0
    while i < len(arr1) and j < len(arr2):
        if arr1[i] < arr2[j]:
            result.append(arr1[i])
            i += 1
        else:
            result.append(arr2[j])
            j += 1
    result += arr1[i:] + arr2[j:]
    return result
```

The asymptotic complexity of this algorithm is: _________________

10) Finding Unique Elements in an Array (2pts)

```
def findUnique(arr):
    unique_elements = set()
    for num in arr:
        unique_elements.add(num)
    return list(unique_elements)
```

The asymptotic complexity of this algorithm is: _________________