Assignment1-Exercise

Name
Student Number

Due time:

23:59 March. 22, 2023

Direction:

Please answer all the questions below and hand in your answers before the due time. All work,must be handed in **on time**.

Questions:

1.Prove (by using the definitions of the notations involved) or disprove (by giving a specific counterexample) the following assertions.

a. If $t(n) \in O(g(n))$, then $g(n) \in \Omega(t(n))$.

b. $\theta(\alpha g(n)) = O(g(n))$, where $\alpha > 0$.

c. $\theta(g(n)) = O(g(n)) \cap \Omega(g(n))$.

d. For any two nonnegative functions $t(n)$ and $g(n)$defined on the set of nonnegative integers, either $t(n) \in O(g(n))$, or $t(n) \in \Omega(g(n))$, or both.

2.Calculate the time complexity of the following algorithms respectively

a.

```
i=0;
while((i+1)*(i+1)<=n)
    i=i+1;
```

b.

```
x=0;
for(i=1;i<=n;i++)
    for(j=1;j<=i;j++)
        for(k=1;k<=j;k++)
            x++;
```

3.Calculate the time complexity of the following recursive algorithms respectively(If it may, the worst, average, and best cases must be investigated separately.)

a.

```
int function(int x, int n) {
    if (n == 0) {
        return 1;
    }
    int t = function(x, n/2);
    if (n % 2 == 1) {
        return t*t*x;
    }
    return t*t;
}
```

b.

```
void Sort(int A[], int low, int high){
    if(low<high){
        int pivot=Partition(A,low,high);
        Sort(A,low,pivot-1);
        Sort(A,pivot+1,high);
    }
}
int Partition(int A[],int low, int high){
    int pivot=A[low];
    while(low<high){
        while(low<high&&A[high]>=pivot) --high;
        A[low]=A[high];
        while(low<high&&A[low]<=pivot) ++low;
        A[high]=A[low];
    }
    A[low]=pivot;
    return low;
}
```

4.Solve the following recurrence relations.

a. $T(n) = T(n-1) + n$ for $n > 0, T(0) = 1$

b. $T(n) = 4T(n-1)$ for $n > 0, T(1) = 5$

c. $T(n) = T(n/3) + n$ for $n > 2, T(1) = 1$ (solve for $n = 3^k$)

Assignment One-Programming

Problem

Consider the situation that Mr. Smith climbs up stairs. He can climb 1 step or 2 steps at one time. It costs him 1 calory for climbing 1 step, and 3 calories for climbing 2 steps at one time.

Q1: Now Mr. Smith needs to climb **m** steps and he only has **n** calories. How many ways are there for him to climb m steps?

Q2: After eating too much, Mr. Smith decides to climb **m** steps for losing weight. Now he also has **n** calories left. At this time, he wants to consume as much calories as possible but no more than **n** calories. How many ways are there for him to climb **m** steps?

Test Cases

Q1 test case 1:

```
input: 6 6
output: 1
```

test case 2:

```
input: 3 6
output: 3
```

test case 3:

```
input: -5 7
output: 0
```

Q2: test case 1:

```
input: 7 6
output: 0
```

test case 2:

```
input: 3 6
output: 2
```

Note

- Please use **C++** to implement above algorithms and provide **screenshots of the output results**

- Algorithms with smaller complexity of time and space consuming are recommended. You can use recursive algorithms to solve them.
- Your program should run **successfully** and output the **correct** answers for every test case.
- Please make sure there are **necessary comments** in your source code. Plagiarism is strictly forbidden.

Submission

- Compilable C++ source codes
- A introduction documentation (PDF is recommended). The document should include algorithm idea and screenshots of running results
- Pack all above files and compress it into a ZIP file. Please rename the ZIP file as 'StudentID_Name_Assignment_1.zip'
- Send the zip file to the email of TA:
- - 2231551@tongji.edu.cn