

2022-CS-111

Question 1

(Estimated Time 20 mins)

(Each proof contains 5 points)

Prove all the thetas (using halving technique or any way you like). Both find out its lower(Omega) and upper(Big O) bound by the same function so that you can declare it to be theta of the given function.

1. $1^2 + 2^2 + 3^2 + 4^2 + \dots + N^2 \sim \Theta(N^3)$
 $1 + 4 + 9 \dots \dots N^2$

So for (N^2)

$j < i$
 So $N^2 \times N \rightarrow \boxed{\Theta(N^3)}$

2. $1 + 2 + 3 + 4 + \dots + N^2 \sim \Theta(N^4)$

$\rightarrow N^2$
 $\rightarrow j < i$
 $\Theta(N^4)$

3. $1 + 3 + 5 + 7 + 9 + \dots + (2N + 1) \sim \Theta(N^2)$

$\frac{(2N+1)}{2} \times \frac{(2N+1)}{2} = \frac{4N^2}{4} = \boxed{\Theta(N^2)}$

4. $2 + 4 + 6 + 8 + \dots + 2N \sim \Theta(N^2)$

$\frac{2N}{2} \times \frac{2N}{2} = \boxed{\Theta(N^2)}$

5. $1 + 2 + 3 + 4 + \dots + (N/2) \sim \Theta(N^2)$

$\frac{N}{2} \times \frac{N}{2} = \boxed{\Theta(\frac{N^2}{4})}$

6. $1 + 2 + 4 + 8 + 16 + \dots + N^2 \sim \Theta(N^2)$

for(int i=0; i < N^2; i++)

{ for(j=0; j < i; j++)

{ -

}

$\Rightarrow N(\log N^2)$

$\Rightarrow \Theta(N^2)$

Question 2

(Estimated Time 100 mins)

(35*2 = 70 Points)

<p>What is the algorithm's complexity of the following piece of code - Sample Solution is in RED.</p> <pre> int Sum=0; // O(1) Time for(int i=0; i<N; i++) // (1+1+1+...+1 --- N Times = O(N) for(int j=0; j<N; j++) Sum++; // (1+1+1+...+1) + (1+1+...+1)+... + (1+1+...+1) added N times // N + N +...+ N = O(N^2) Overall Complexity : O(1) + O(N) + O(N^2) + O(N^2) = O(N^2) </pre>	<p>2) What is the algorithm's complexity of the following piece of code</p> <pre> int Sum=0; for(int i=0; i<N; i++) Sum++; for(int j=0; j<N; j++) Sum++; </pre> <p style="text-align: right;">$N+N$ $= O(N)$</p>
<p>3)</p> <p>What is the algorithm's complexity of the following piece of code</p> <pre> int Sum=0; for(int i=0; i<N; i++) → O(N) for(int j=0; j<N; j++) → O(N) for(int k=0; k<N; k++) O(N) Sum++; </pre> <p style="text-align: right;">$\rightarrow O(N^3)$</p> <pre> for(int i=0; i<N; i++) for(int j=0; j<N; j++) for(int k=0; k<N; k++) Sum++; </pre> <p style="text-align: right;">$\rightarrow O(N^3)$</p>	<p>4)</p> <p>What is the algorithm's complexity of the following piece of code</p> <pre> int Sum=0; for(int i=0; i<N; i++) Sum++; for(int j=0; j<N; j++) Sum++; for(int k=0; k<N; k++) Sum++; for(int m=0; m<N; m++) Sum++; for(int n=0; n<N; n++) Sum++; for(int p=0; p<N; p++) Sum++; </pre> <p style="text-align: right;">$O(N) \text{ or } O(6N)$</p>
<p>5)</p> <pre> int Sum=0; for(int i=0; i<N; i++) N for(int j=0; j<i; j++) N for(int k=0; k<j; k++) N Sum++; </pre> <p style="text-align: right;">$O(N^3)$</p>	<p>6</p> <pre> int Sum=0; for(int i=0; i<N; i+=2) → O(N/2) for(int j=0; j<i; j+=2) → O(N/2) for(int k=0; k<j; k+=2) → O(N/2) Sum++; </pre> <p style="text-align: right;">$O(\frac{N^3}{8})$</p>
<p>7</p> <pre> int Sum=0; for(int i=1; i<N; i*=2) log N for(int j=1; j<N; j*=2) log N Sum++; </pre> <p style="text-align: right;">$O(\log^2 N)$</p>	<p>8</p> <pre> int Sum=0; for(int i=1; i<N; i*=2) log N Sum++; for(int j=1; j<N; j*=2) log N Sum++; </pre> <p style="text-align: right;">$O(2\log N)$</p>

<p>9</p> <pre>for(int i=1; i<=N*N; i+=2) for(int j=1; j<N*N; j*=2) Sum++;</pre> <p>$\frac{N^2}{2} \log_2 N^2$ $O(N^2 \log N^2)$</p>	<p>10</p> <pre>for(int i=1; i<=N*N; i+=2) Sum++; for(int j=1; j<N*N; j*=2) Sum++;</pre> <p>$(\frac{N^2}{2}) \log N^2$ $O(N^2) \text{ Ans}$</p>
<p>11</p> <pre>for(int i=1; i<=N*N; i*=2) for(int j=1; j<N*N; j*=2) Sum++;</pre> <p>$(\log N^2)$ $(\log N^2)$ $(\log^2 N^2)$</p>	<p>12</p> <pre>for(int i=1; i<=N*N; i*=2) Sum++; for(int j=1; j<N*N; j*=2) Sum++;</pre> <p>$\log N^2$ $\log N^2$ $O(\log N^2)$</p>
<p>13</p> <pre>int Sum=0; for(int i=1; i<=N; i*=2) for(int j=1; j<=N; j*=2) for(int k=1; k<=N; k*=2) Sum++;</pre> <p>$(\log N)$ $(\log N)$ $(\log N)$ $O(\log N)^3$</p>	<p>14</p> <pre>int Sum=0; for(int i=1; i<=N; i*=2) Sum++; for(int j=1; j<=N; j*=2) Sum++; for(int k=1; k<=N; k*=2) Sum++;</pre> <p>$(\log N)$ $(\log N)$ $(\log N)$ $(3 \log N)$</p>
<p>15</p> <pre>int sum,i,j; sum = 0; for (i=1; i<n; i=i*2) { for (j=0; j<n; ++j) { sum++; } }</pre> <p>$(\log_2 N)$ N $O(N \log_2 N)$ $O(N) \text{ Ans}$</p>	<p>16</p> <p><u>BE CAREFUL GEOMETRIC SERIES</u></p> <pre>int sum,i,j; sum = 0; for (i=1; i<n; i=i*2) { for (j=0; j<i; ++j) { sum++; } }</pre> <p>$(\log N)$ N $O(N \log N)$ $O(N) \text{ Ans}$</p>
<p>17</p> <p><u>BE CAREFUL GEOMETRIC SERIES</u></p> <pre>int sum,i,j; sum = 0; for (i=1; i<n; i=i*5) { for (j=0; j<i; j+=2) { sum++; } }</pre> <p>$(\log_5 N)$ $\frac{N}{2}$ $(\frac{N}{2} \log_5 N) \Rightarrow O(N)$</p>	<p>18</p> <pre>int sum,i,j; sum = 0; for (i=1; i<n; i=i*4) { for (j=0; j<n; j+=3) { sum++; } }</pre> <p>$(\log_4 N)$ $\frac{N}{3}$ $O(N/3) \text{ Ans}$</p>

What will be the output (the value of Sum) of the program asymptotically in BIG-O notation, I am asking here the complexity of loop rather the asymptotic bound on the value of Sum:

```
int Sum = 0;
for(int i=1; i<=n; i+=1)
{
    Sum+=i;
}
cout<<Sum<<endl;
```

$\rightarrow O(N)$
 $\rightarrow \frac{n(n+1)}{2}$
 $O(N^2)$

21 What is the time complexity of the algorithm:

```
int Sum = 0;
for(int i=1; i<=n; i+=1)
{
    for(int j=1; j<=i; j++)
    {
        Sum++;
    }
}
cout<<Sum<<endl;
```

$O(N)$
 N
 $O(N^2)$

40* Complexity of primeNumber function.

```
int sqrt(int N)
{
    int d;
    for(d=0; d*d<=N; d++) { }
    return d-1;
}

bool primeNumber(int n)
{
    bool isPrime = true;
    int lmt = (sqrt(n));
    for (int d=2; d <= lmt; ++d)
    {
        if (n%d==0)
            return false;
    }
    return true;
}
```

\sqrt{N}
 $O(\sqrt{N})$

20 What will be the output (the value of Sum) of the program asymptotically in BIG-O notation:

```
int Sum = 0;
for(int i=1; i<=n; i*=2)
{
    Sum+=i;
}
cout<<Sum<<endl;
```

$(\frac{N^2}{2})$ $O(N^2)$

22 What is the time complexity of the algorithm:

```
int Sum = 0;
for(int i=1; i<=n; i*=2)
{
    for(int j=1; j<=i; j++)
    {
        Sum++;
    }
}
cout<<Sum<<endl;
```

$(\log N)$
 N
 $O(N \log N)$

41* Complexity of primeNumber function:

```
int sqrt(int N)
{
    int d;
    for(d=0; d*d<=N; d++) { }
    return d-1;
}

bool primeNumber(int n)
{
    bool isPrime = true;
    for (int d=2; d <= sqrt(n); ++d)
    {
        if (n%d==0)
            return false;
    }
    return true;
}
```

\sqrt{N}
 $\sqrt{N} \times \sqrt{N}$
 $O(N)$
 $O(\sqrt{N})$

23 What is the time complexity of the algorithm:

int f1(int n)

```
{
    int K=0;
    for(int j=0; j*j<=n*n; j++) K++;
    return K;
}
```

int main()

```
{
    int Sum = 0, n;
    cin >> n;
    for(int i=1; i<=f1(n); i+=1)
        for(int j=1; j<=i; j++) Sum++;
    cout << Sum << endl;
}
```

$O(N^3)$

24

What is the time complexity of the algorithm:

int f1(int n)

```
{
    int K=0;
    for(int j=1; j*j<=n; j*=2) K++;
    return K;
}
```

int main()

```
{
    int Sum = 0;
    int n;
    cin >> n;
    for(int i=1; i<=f1(n); i+=1)
        for(int j=1; j<=i; j++) Sum++;
    cout << Sum << endl;
}
```

$O(N^2 \sqrt{N})$

25

What is the time complexity of the algorithm:

int f1(int n)

```
{
    int K=0;
    for(int j=1; j*j<=n; j++) K++;
    return K*K;
}
```

int main()

```
{
    int Sum = 0;
    int n;
    cin >> n;
    int Terminator = f1(n);
    for(int i=1; i<= Terminator; i+=1)
    {
        for(int j=1; j<=i; j++)
        {
            Sum++;
        }
    }
    cout << Sum << endl;
}
```

$O(N^4)$

26

What is the time complexity of the algorithm:

int f1(int n)

```
{
    int K=0;
    for(int j=0; j*j<=n; j++) K++;
    return K;
}
```

int main()

```
{
    int Sum = 0;
    int n;
    cin >> n;
    int Terminator = f1(n);
    for(int j=1; i<= Terminator; i+=1)
    {
        for(int j=1; j<=i; j++)
        {
            Sum++;
        }
    }
    cout << Sum << endl;
}
```

$O(N)$

27

for (i=1; i<n; i=i*4) $\rightarrow (\log_4 N)$

```
{
    cout << i;
    for (j=0; j<n; j=j+2)  $\rightarrow \frac{n}{2}$ 
    {
        cout << j;
        sum++;
    }
    cout << sum;  $\rightarrow O(N)$ 
}
```

28

for (i=1; i<n; i=i*4) $\rightarrow (\log_4 N)$

```
{
    cout << i;
    for (j=0; j<i; j=j+2)  $\rightarrow (\frac{n}{2})$ 
    {
        cout << j;
        sum++;
    }
    cout << sum;  $\rightarrow O(\frac{N}{2})$ 
}
```

for (i=1; i<=n*n; ++i) n^2

```
cout << i;
Sum=0;
for (j=1; j<=i; ++j)
{
    Sum++;
    cout << i;
}
cout << Sum;
```

$O(n^4)$

30

for (i=1; i<=n*n*n; ++i)

$O(n^3)$

```
{
    cout << i;
    Sum=0;
    for (j=1; j<=i; ++j)
    {
        Sum++;
        cout << i;
    }
    cout << Sum;
```

$O(n^6)$

31

for (i=1; i<=n*n*n; i*=2)

$(\log_2 n^3)$

```
{
    cout << i;
    Sum=0;
    for (j=1; j<=i; j++)
    {
        Sum++;
        cout << i;
    }
    cout << Sum;
```

n^3

$n^3(\log_2 n^3)$

$O(n^3)$

32

for (i=1; i<=n*n*n; i*=2) $(\log n^3)$

```
{
    cout << i;
    Sum=0;
    for (j=1; j<=n; j++)  $\leftarrow n^3$ 
    {
        Sum++;
        cout << i;
```

} for (k=1; k<=n; k++) $\leftarrow n^3$

```
{
    Sum++;
    cout << i;
}
cout << Sum;
```

$O(n \log n^3)$

33

for (i=1; i<=n*n*n; i*=2)

$(\log n^3)$

```
{
    cout << i;
    Sum=0;
    for (j=1; j<=i; j++)
    {
        Sum++;
        cout << i;
```

n^3

```
for (j=1; j<=n; j*=2)  $(\log n)$ 
{
    Sum++;
    cout << i;
```

$(\log n)$

```
cout << Sum;
```

$O(n^3)$

34

for (i=1; i<=n*n*n; i*=2)

$\log n^3$

```
{
    cout << i;
    Sum=0;
    for (j=1; j<=i; j++)
    {
        Sum++;
        cout << i;
```

n^3

```
for (j=1; j<=n; j++)  $n$ 
{
    Sum++;
    cout << i;
```

n

```
cout << Sum;
```

$O(n^3)$

35-36

```

for (int i=1; i <= n; i = i * 2)
{
    for (j = 1; j <= i; j = j * 2)
        cout << " ";
}

```

$(\log n)$
 $(\log n)$
 $(\log^2 n)$

```

for (int i=1; i <= n; i = i * 2)
    for (j = 1; j <= i; j = j * 2)
        cout << " ";

```

$\rightarrow \log^2 n$

```

for (int i=1; i <= n; i = i * 2)
    for (j = 1; j <= i; j = j * 2)
        cout << " ";

```

$\rightarrow \log^2 n$
 $O(2 \log^2 n)$

37

```

for (i=0; i < n; i=i+3)
{
    cout << i;
    for (j=1; j < n; j=j*3)
    {
        cout << j;
        sum++;
    }
    for (k=1; k < n; k=k*3)
    {
        cout << j;
        sum++;
    }
    cout << sum;
}

```

$\rightarrow n/3$
 $(\log n)$
 $(\log n)$
 $O(n)$

38

```

for (int i=1; i <= n; i = i * 2)
{
    for (j = 1; j <= i; j = j * 2)
    {
        cout << " ";
    }
}

```

$(\log^2 n)$

```

for (int i=0; i <= N; i++)
{
    Sum++;
}

```

$\rightarrow n$
 $O(n)$

39

```

for (i=0; i < n; i=i+3)
{
    cout << i;
    for (j=1; j < n; j=j*3)
    {
        sum++;
    }
}
for (k=1; k < n; k=k*3)
{
    cout << j;
    sum++;
}
cout << sum;

```

$(n/3)$
 $(\log 3)$
 $\log 10$
 $O(n)$
 ~~$O(n/3)$~~
 ~~$O(\log n)$~~

Question 3 Analyze the complexity Θ of the following functions in terms of N . 1*5+5 = 10 points

<pre> int f1(int N) { int Count = 0; for (int i = 1; i <= N; i *= 2) for (int j = 1; j <= i; j++) Count++; return Count; } </pre> <p>$(\log n)$ n $O(n)$</p>	<pre> int f2(int N) { int Count = 0; int C = f1(N); for (int i = 0; i < C; i++) Count++; return Count; } </pre> <p>$\rightarrow O(n)$ $O(n)$</p>	<pre> int f5(int N) { int Count = 0; for (int i = 0; i < sqrt(f1(N) * f1(N)); i++) Count++; return Count; } </pre> <p>n $O(n)$</p>
<pre> int f3(int N) { int Count = 0; int C = sqrt(f1(N)); for (int i = 1; i < C; i *= 2) Count++; return Count; } </pre> <p>$(\log_2 \sqrt{n})$</p>	<pre> int f4(int N) { int Count = 0; for (int i = 0; i < f1(N) * f1(N); i++) Count++; return Count; } </pre> <p>n^2 $O(n^2)$</p>	<pre> int Sum = 0; int f6(int N) { if (N == 1) return 1; Sum += f1(N); Sum += f2(N); Sum += f3(N); Sum += f4(N); Sum += f5(N); return Sum; } </pre> <p>$O(n)$ $O(n^2)$</p>