Data Structure

Fall 2019

M 16:00-18:00 W 11:00-13:00

http://smart.hallym.ac.kr

Instructor: **Jin Kim**
010-6267-8189(033-248-2318)
jinkim@hallym.ac.kr

Office Hours:

Lab22(Complexity)
시간복잡도

Spring 2018

http://smart.hallym.ac.kr

Instructor: Jin Kim

010-6267-8189(033-248-2318)

jinkim@hallym.ac.kr

Office Hours:

Objectives(실습목표)

- 자바의 실행시간 계산 방법을 이해한다.
- 시간복잡도를 이해한다.
- 빠른 알고리즘이 가장 좋은 알고리즘임을 이해한다.

Measuring Elapsed Time(실행시간측정)

```
1  import java.util.*;
2
3  public class ElapsedTime {
4
5      public static void main(String[] args) {
6          try {
7              long start = System.currentTimeMillis();
8              System.out.println(new Date());
9              Thread.sleep(5*1000);
10             System.out.println(new Date());
11             long end = System.currentTimeMillis();
12             long diff = end - start;
13             System.out.println("Difference: " + diff);
14         } catch (Exception e) {
15             System.out.println("An exception occurs");
16         }
17     }
18 }
```

Measuring Elapsed Time(실행시간측정)

```
1  import java.util.*;
2
3  public class ElapsedTime {
4
5      public static void main(String[] args) {
6          try {
7              long start = System.currentTimeMillis();
8              System.out.println(new Date());
9              Thread.sleep(5*1000);
10             System.out.println(new Date());
11             long end = System.currentTimeMillis();
12             long diff = end - start;
13             System.out.println("Difference: " + diff);
14         } catch (Exception e) {
15             System.out.println("An exception occurs");
16         }
17     }
18 }
```

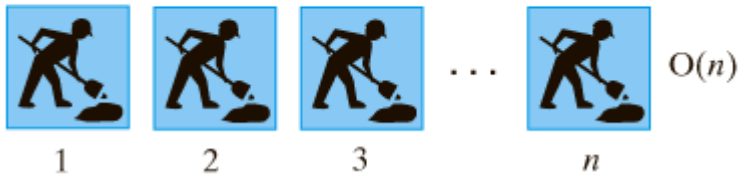
5초동안 멈춰 있어라

5000마이크로세컨드, 즉 5초동안 잠자고 있음. 실행시간은 대략 5000이 출력될 것임

Assignment 1

- 1부터 n까지의 합을 구하는 프로그램(sum1.java)을 작성하라. 아래의 알고리즘을 사용하라. N=백만, 2백만, …..,천만일때의 시간을 계산하라.

```
for i = 1 to n
   sum = sum + i
```



Sum1.java
1부터 n까지의 합 O(n)

```
1  import java.util.*;
2
3  public class ElapsedTime {
4
5      public static void main(String[] args) {
6          try {
7              long start = System.currentTimeMillis();
8              System.out.println(new Date());
9              Thread.sleep(5*1000);
10             System.out.println(new Date());
11             long end = System.currentTimeMillis();
12             long diff = end - start;
13             System.out.println("Difference: " + diff);
14         } catch (Exception e) {
15             System.out.println("An exception occurs");
16         }
17     }
18 }
```

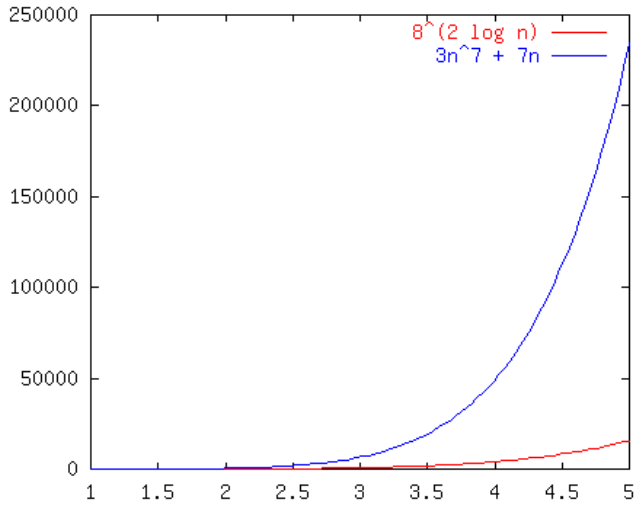
```
for(int i=1; i<=n; i++){
    sum= sum+i;
}
```

교체

Assignment2 O(n^2)

- 1부터 n까지의 합을 구하는 프로그램(sum2.java)을 작성하라. 아래의 알고리즘을 사용하라. N=백만, 2백만, …..,천만일때의 시간을 계산하라. 두 알고리즘의 실행시간을 엑셀을 사용하여 선을 사용한 그래프(비교1.xls)로 그려라. 어떤 알고리즘이 빠른가? 붉은 색 코드는 실행시간이 더 소요되게 만들지만, 값에는 영향없음.

```
for i = 1 to n {
        for j = 1 to n {
                sum = sum + 1;
        }
}
```

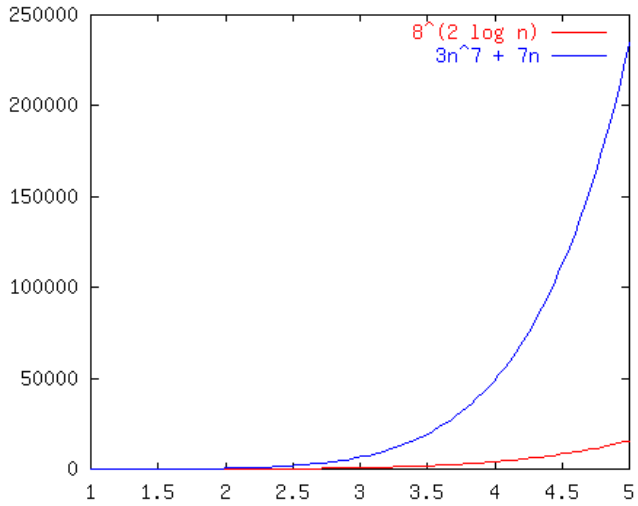


- N 제곱알고리즘

Assignment3 O(n^3)

- 1부터 n까지의 합을 구하는 프로그램(sum3.java)을 작성하라. 아래의 알고리즘을 사용하라. N=백만, 2백만, …..,천만일때의 시간을 계산하라. 세 알고리즘의 실행시간을 엑셀을 사용하여 선을 사용한 그래프(비교1.xls)로 그려라. 어떤 알고리즘이 빠른가? 붉은 색 코드는 실행시간이 더 소요되게 만들지만, 값에는 영향없음.

```
for i = 1 to n {
    for j = 1 to i {
        sum = sum + 1;
        for k = 1 to j {
            sum = sum +  0; // 0을 더한 이유는 sum 의 오버플로우 방지
        }
    }
}
```



N 세제곱알고리즘

SumTotal.java

- 위의 모든 프로그램들을 한 프로그램으로 만들어 업로드하라.

Assignment 5

- Task: Compute the x^n for an integer $n = 2^k$
- X=2로하자. 즉 2^n을 구하는 프로그램을 작성함

```
int prod=1; x=1;                    //n=1,2,4,8,16, 32를 차례대로 넣어라

 for(i=1;i<=n;i++){
    prod=prod*x;
}
```

- The efficiency is O(n) : 곱셈을 거의 n번함. N에 비례하여 실행시간이 소요됨. 이때 n을 증가시키면 prod값이 커져 오버플로우가 발생할 것임.
- X=1로 바꾸어 실행하라, n=10만, 20만,30만,…일때 실행시간을 계산하라(prod3.java).
- (x=1로 한 이유는 오버플로우때문)

Assignment 6

- Task: Compute the x^n for an integer $n = 2^k$
  - X=2로하자. 즉 2^n을 구하는 프로그램을 작성함

```
int prod=2;                    //n=1,2, 3 를 차례대로 넣어라
  for (i=1;i<=n;i++) {
      prod=prod*prod;//
}
// 이 프로그램은 2^2 ,2^4 ,2^8램은… 을 계산할 것이다.
```

  - The efficiency is O(logn) : 곱셈을 거의 n번함.
  - $x^2 = x^1 * x^1$, $x^4 = x^2 * x^2$, $x^8 = x^4 * x^4$,…, x^n=?
  - X=1, n=10만, 20만,30만,…에 접근할때 실행시간을 계산하라(prod4.java). (x=1로 한 이유는 오버플로우때문)
  - K는 어떻게 계산할래?
  - 두 알고리즘의 실행시간을 엑셀을 사용하여 선을 사용한 그래프(비교2.xls)로 그려라. 어떤 알고리즘이 빠른가?

Assignment 6

- Task: Compute the x^n for an integer $n = 2^k$
  - X=2로하자. 즉 2^n을 구하는 프로그램을 작성함

```
int prod=2;                    //k=1,2, 3 를 차례대로 넣어라
  for (i=1;i<=k;i++) {
      prod=prod*prod;//
}
// 이 프로그램은 2^2 ,2^4 ,2^8램은… 을 계산할 것이다.
```

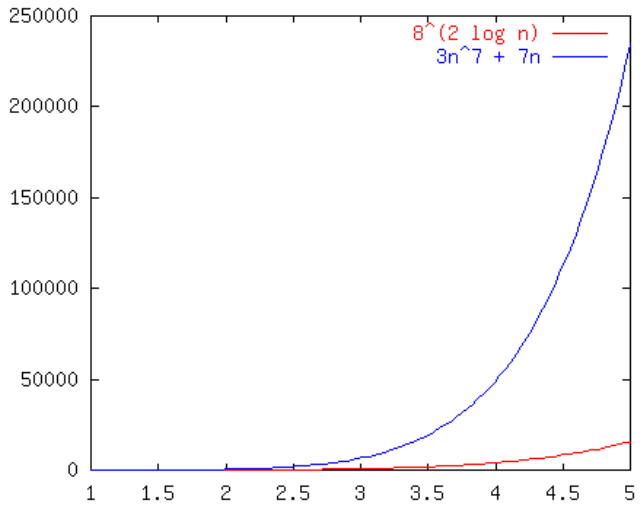
  - The efficiency is O(logn) : 곱셈을 거의 n번함.
  - $x^2 = x^1 * x^1$, $x^4 = x^2 * x^2$, $x^8 = x^4 * x^4$,…, x^n=?
  - X=1, n=10만, 20만,30만,…에 접근할때 실행시간을 계산하라(prod4.java). (x=1로 한 이유는 오버플로우때문)
  - 두 알고리즘의 실행시간을 엑셀을 사용하여 선을 사용한 그래프(비교2.xls)로 그려라. 어떤 알고리즘이 빠른가?

ProdTotal.java

- 위의 모든 프로그램들을 한 프로그램으로 만들어 업로드하라.

그래프의 모양

- N=1, 2, 4, 8, 16, 32….에 대하여 두 프로그램의 실행시간을 계산하고 그래프를 그린다.



Sort.java

- 임의의 양의 정수 n개를 무작위로 생성한 후 정렬하는 프로그램을 작성하고, 실제 실행 시간을 구하라. 이때선택정렬을 사용하라.

Zip all your programs(name.zip) and upload to smart.hallym.ac.kr