|  |
| --- |
| **[511643] 자료구조** |
| **실습 #03 보고서** |

|  |  |
| --- | --- |
| **이름** | 조아현 |
| **학번** | 20215247 |
| **소속**  **학과/대학** | 소프트웨어융합대학  빅데이터학과 |
| **분반** | 01 (담당교수: 김태운) |

## <주의사항>

* 개별 과제 입니다. (팀으로 진행하는 과제가 아니며, 모든 학생이 보고서를 제출해야 함)
* 각각의 문제 바로 아래에 답을 작성 후 제출해 주세요.
  + 소스코드/스크립트 등을 작성 한 경우, 해당 파일의 이름도 적어주세요.
* SmartLEAD 제출 데드라인:
  + 2주 뒤 실습시간 전날(다음 다음번 실습 전날) 23:55까지
  + 데드라인을 지나서 제출하면 0점(예외 없음)
  + 주말/휴일/학교행사 등으로 인한 데드라인 연장 없음
  + 부정행위 적발 시, 원본(보여준 사람)과 복사본(베낀 사람) 모두 0점 처리함
* SmartLEAD에 아래의 파일을 제출 해 주세요
  + 보고서(PDF 파일로 변환 후 제출 권장하나, WORD 형식으로 제출도 가능)
  + 보고서 파일명에 이름과 학번을 입력 해 주세요.
  + 소스코드, 스크립트, Makefile 등을 작성해야 하는 경우, 모든 파일을 하나의 zip 파일로 압축하여 제출(또는 본 문서에 소스코드 화면 캡쳐해서 붙여넣기)

## <개요>

이번 과제는 실행시간을 측정하는 과제입니다.

## <실습 과제>

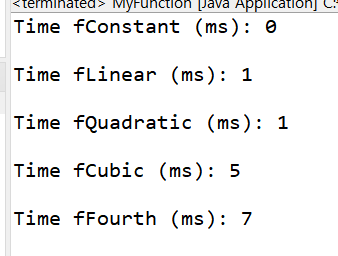
|  |
| --- |
| **[Q 0] 요약 [배점: 10]**  이번 과제에서 배운 내용 또는 과제 완성을 위해서 무엇을 했는지 2~3문장으로 요약하세요. |

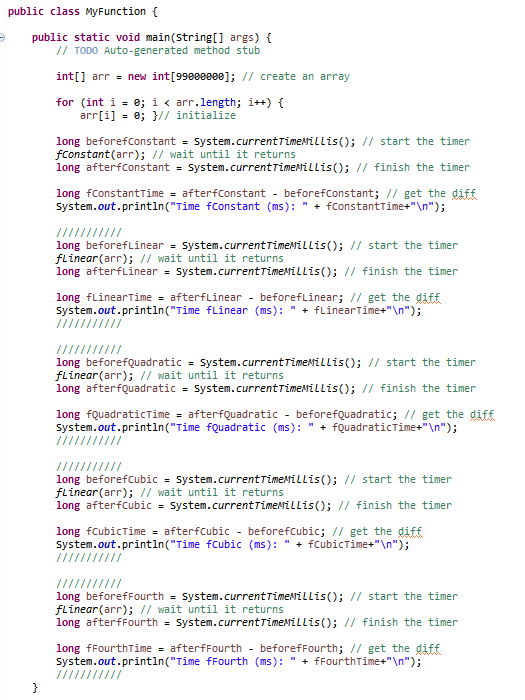
답변:

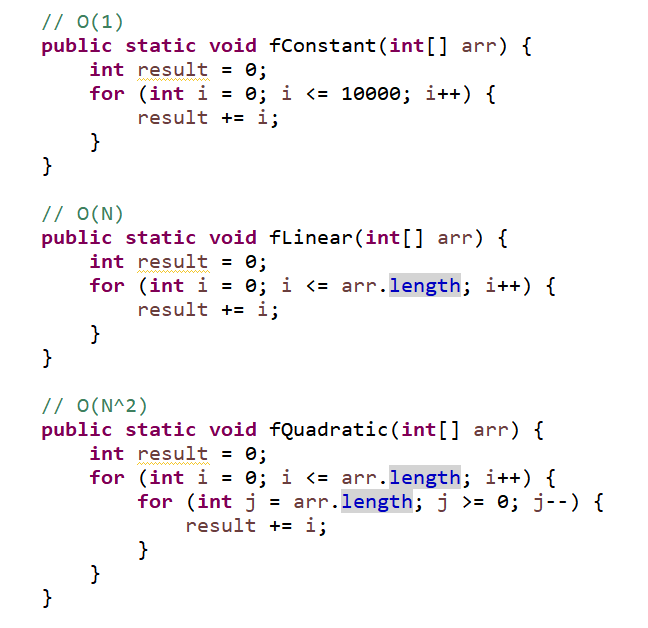
알고리즘이 수행하는 기본 연산 횟수를 입력 크기에 대한 함수로 표현하는 수행 시간에 대하여 새로 알게 되었습니다. 과제를 수행하며 식을 보고서는 최고 차수 항만을 보고 O (Big-Oh) 표기를 할 수 있지만, 코드를 O (Big-Oh) 표기법으로 어떻게 나타내야 하는지 어려워 강의 자료를 많이 참고하였습니다.

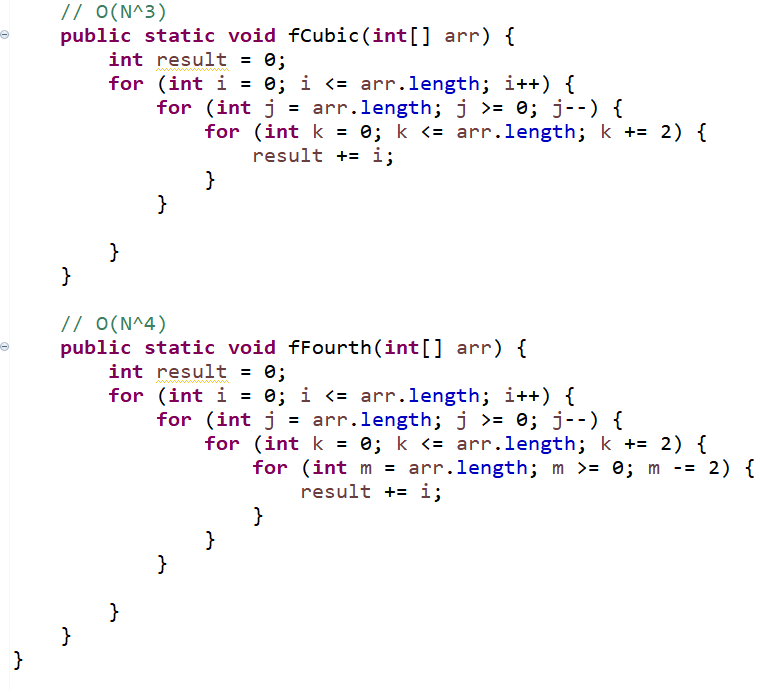
|  |
| --- |
| **[Q 1] 실행시간 [배점: 30]**  함수의 실행 시간을 측정하는 방법은 일반적으로 다음과 같습니다. 예를 들어 foo() 라는 함수의 실행 시간을 측정하려고 한다면,  <코드>  (초기화 등등...)  t1 = 현재시간  foo()함수 호출... >> 리턴  t2 = 현재시간  함수\_실행에\_소요된\_시간 = t2 - t1 |
| **fMyFunction 함수의 실행 시간 측정하기**  **int**[] arr = **new** **int**[500]; // create an array  **for**(**int** i = 0 ; i < arr.length ; i++ ) arr[i] = 0; // initialize  **long** beforeTime = System.*currentTimeMillis*(); // start the timer  *fMyFunction*(arr); // wait until it returns  **long** afterTime = System.*currentTimeMillis*(); // finish the timer  **long** diffTimeMilliSecond = afterTime-beforeTime; // get the diff  System.***out***.println("Time diff (ms): " + diffTimeMilliSecond); |
| 위의 코드는 fMyFunction 함수를 실행하는데 소요된 시간을 출력합니다.   1. O(1) 상수 시간이 소요되는 fConstant(int[] arr) 함수를 구현하세요. 함수 실행 시 상수 시간이 소요되기만 한다면 어떻게 구현하든 상관 없습니다. 시간을 측정해서 터미널에 출력하세요. 2. O(N) 선형 시간이 소요되는 fLinear(int[] arr) 함수를 구현하세요. 시간을 측정해서 터미널에 출력하세요. 여기서 N = arr.length 입니다. 3. O(N^2) 제곱 시간이 소요되는 fQuadratic(int[] arr) 함수를 구현하세요. 함수 실행 시 제곱 시간이 소요되기만 한다면 어떻게 구현하든 상관 없습니다. 시간을 측정해서 터미널에 출력하세요. 여기서 N = arr.length 입니다. 4. O(N^3) 세제곱 시간이 소요되는 fCubic(int[] arr) 함수를 구현하세요. 함수 실행 시 세제곱 시간이 소요되기만 한다면 어떻게 구현하든 상관 없습니다. 시간을 측정해서 터미널에 출력하세요. 여기서 N = arr.length 입니다. 5. O(N^4) 네제곱 시간이 소요되는 fFourth(int[] arr) 함수를 구현하세요. 함수 실행 시 네제곱 시간이 소요되기만 한다면 어떻게 구현하든 상관 없습니다. 시간을 측정해서 터미널에 출력하세요. 여기서 N = arr.length 입니다.   터미널 출력결과를 캡처하고 본 문서에 첨부하세요. 소스코드도 첨부파일로 제출하세요.  \*\* 참고: 함수가 어떤 기능을 수행하는지는 중요하지 않습니다. 자유롭게 구현하세요. 실행 시간이 너무 오래 걸리면 배열의 크기를 줄여서 사용하세요. 실행 시간이 매우 짧고 위의 다섯 가지 경우에서 시간 차이가 거의 없다면 배열의 크기를 늘리세요. |

답변:



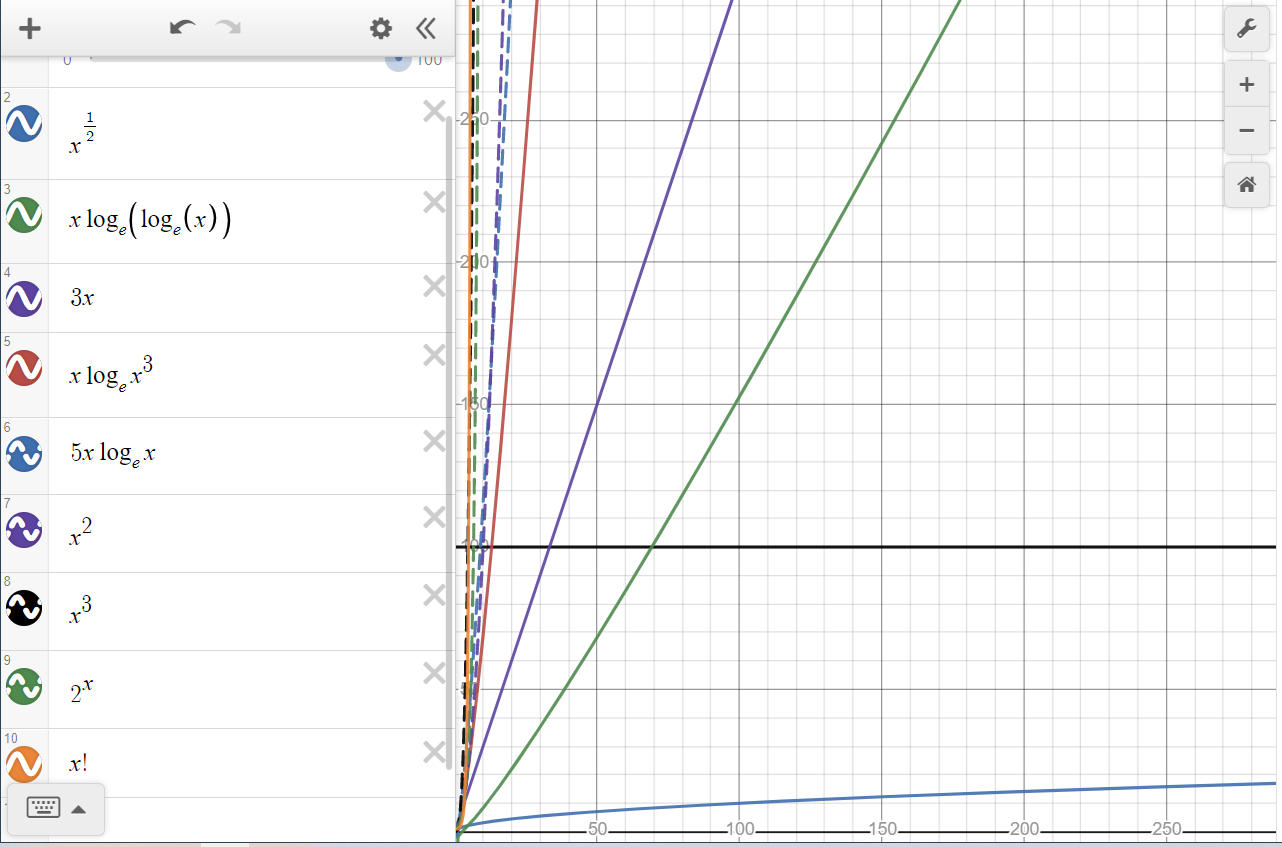






|  |
| --- |
| **[Q 2] 시간 복잡도 계산 1 [배점: 20]**  문제 1) <https://www.desmos.com/calculator> 웹 페이지에서 아래의 함수를 모두 입력하고, 화면을 캡처해서 아래에 첨부하세요.  문제 2) 아래의 함수들을 증가율 순으로 오름차순 정렬하세요. 즉, 가장 낮은 시간 복잡도를 가지는 함수를 먼저 적고, 가장 큰 시간 복잡도를 가지는 함수를 마지막에 적으세요.  N1/2, 5NlogeN, NlogelogeN, 3N, 2N, N3, N!, 100, NlogeN3, N2 |

답변 1)



답변 2)

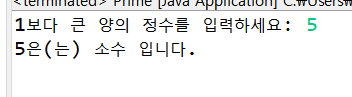
100, N1/2, NlogelogeN, 3N, NlogeN3, 5NlogeN, N2, N3, 2N, N!

|  |  |  |
| --- | --- | --- |
| **[Q 3] 시간 복잡도 계산 2 [배점: 20]**  다음 각각에 대해서 참(T) 인지 거짓(F)인지 답하세요 | | |
| **문제 번호** | **문제** | **답변** |
| **3.1** | 2N2 + 3N3 = O(N2) | T |
| **3.2** | 2N2 + 3N3 = Ω(N2) | T |
| **3.3** | 2N2 + 3N3 = Θ(N2) | F |

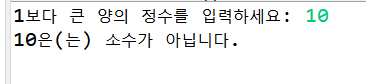
*답변: 위 테이블에 답변을 작성하세요*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **[Q 4] 소수 찾기 [배점: 20]**  소수 (Prime Number)는 1과 자기 자신으로만 나누어 떨어지는 수라고 정의된다. 소수의 정의에 따라 소수를 찾는 프로그램을 작성하시오(=즉, 1과 자기자신을 제외한 숫자 중에서 나누어 떨어지는 수가 있는지를 순차적으로 검색하는 방법으로 프로그램을 구현할 것). 소수를 찾는 영리한 알고리즘이 있지만, 소수의 정의에 기반하여 프로그램을 작성하세요. 콘솔 화면에서 사용자 입력(정수)을 받고, 소수인지 아닌지 그 결과를 알려주면 됩니다.   |  |  | | --- | --- | | **실행 예: 사용자가 5를 입력한 경우** | **실행 예: 사용자가 10을 입력한 경우** | | <콘솔 출력 예시>  1보다 큰 양의 정수를 입력하세요: 5  5은(는) 소수 입니다. | <콘솔 출력 예시>  1보다 큰 양의 정수를 입력하세요: 10  10은(는) 소수가 아닙니다. |   문제 1) 사용자 입력이 5인 경우, 콘솔 출력을 캡처해서 아래에 첨부하세요.  문제 2) 사용자 입력이 10인 경우, 콘솔 출력을 캡처해서 아래에 첨부하세요.  문제 3) 프로그램의 복잡도를 Big Oh 표기법으로 나타내시오. |

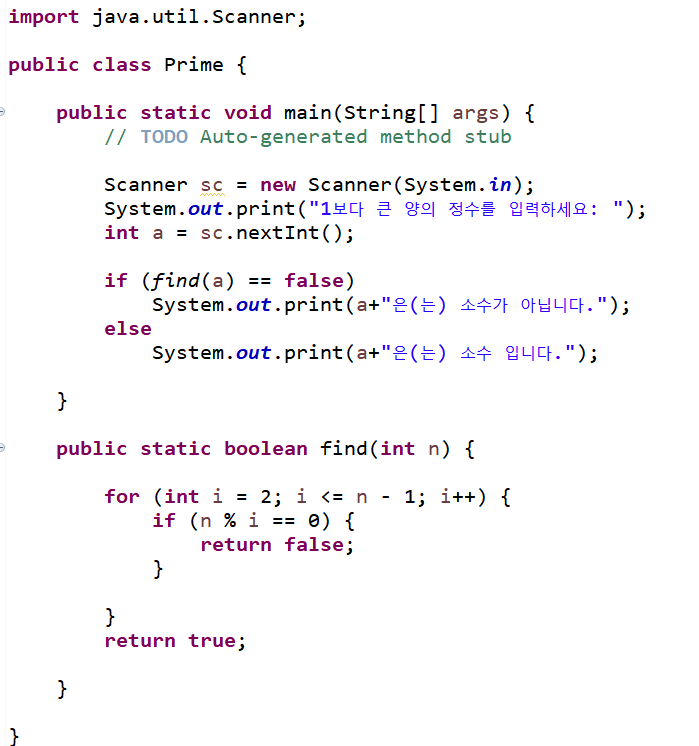
답변 1)



답변 2)



답변 3) O(N)

****

**끝! 수고하셨습니다 ☺**