TUGAS 2 PRAKTIKUM

KOMPLEKSITAS WAKTU DARI ALGORITMA



Muhammad Fadli Muflih

140810170006

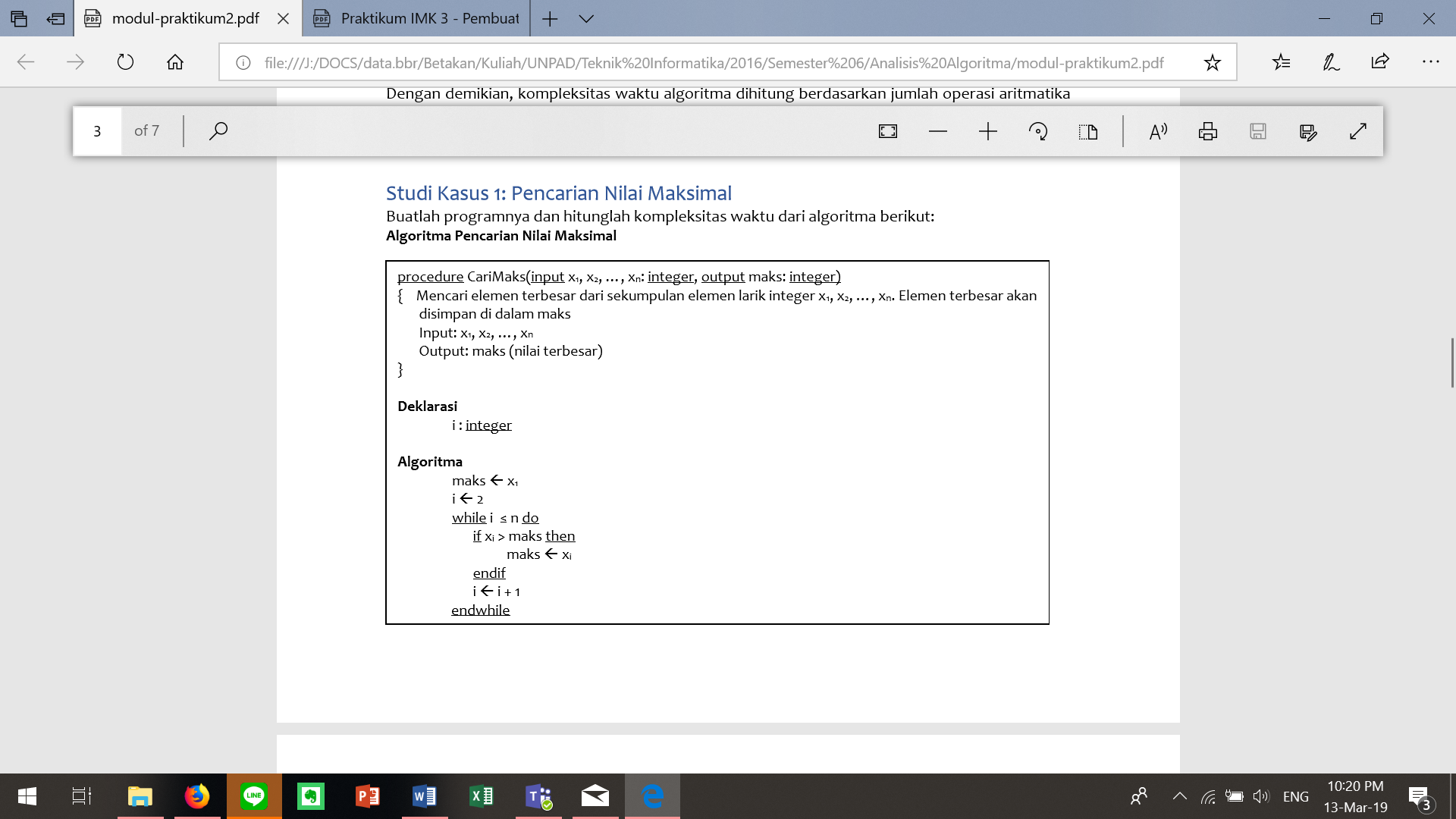
PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

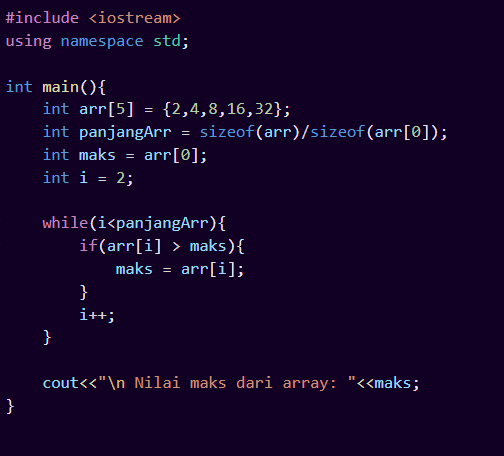
2019

**Nomor 1**

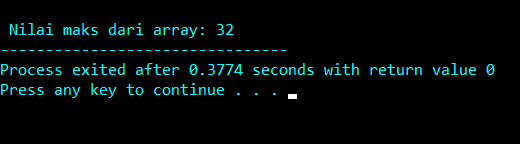


Jawab:

* Source code



* Hasil screenshoot



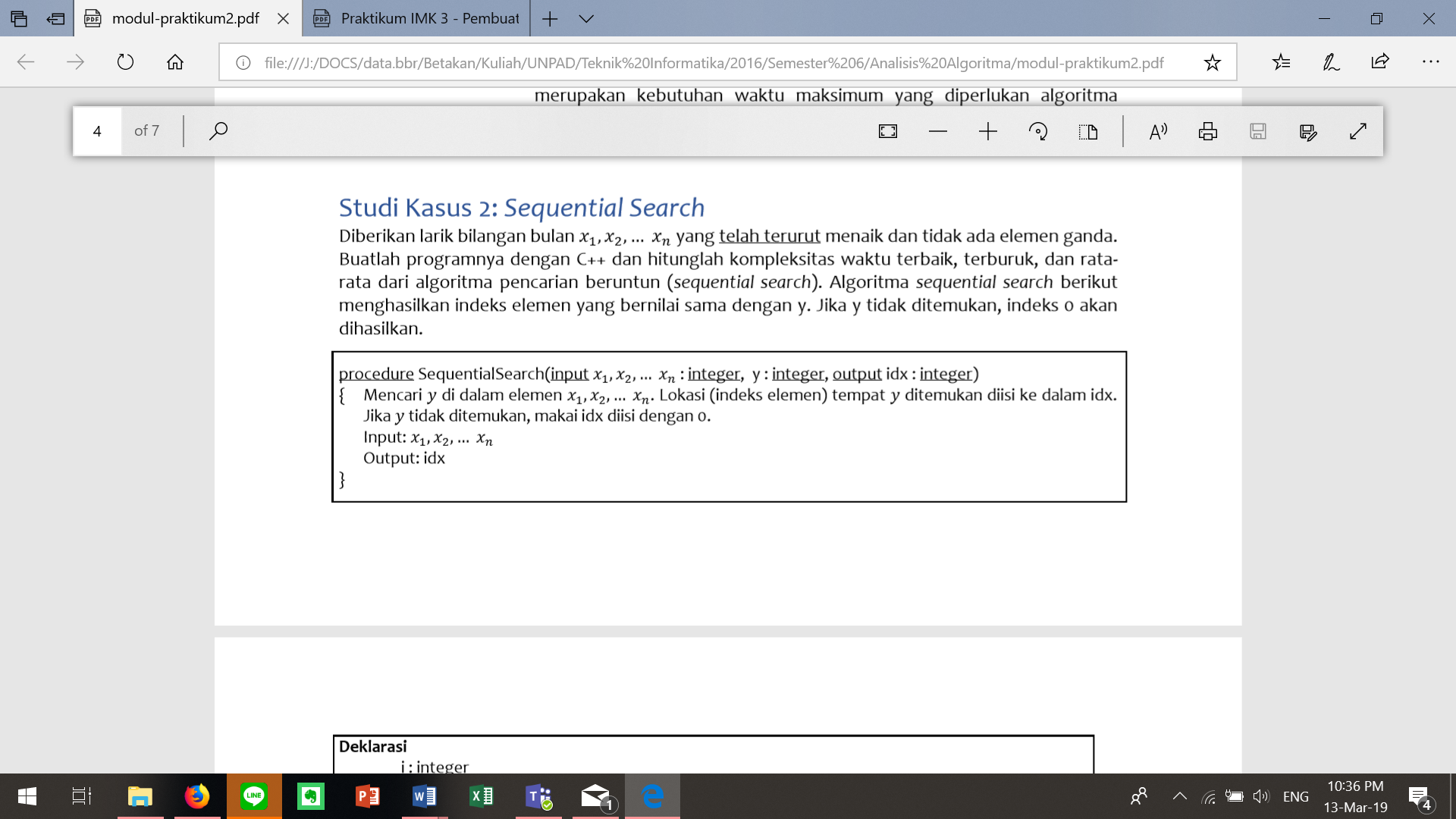
* Kompleksitas waktu:

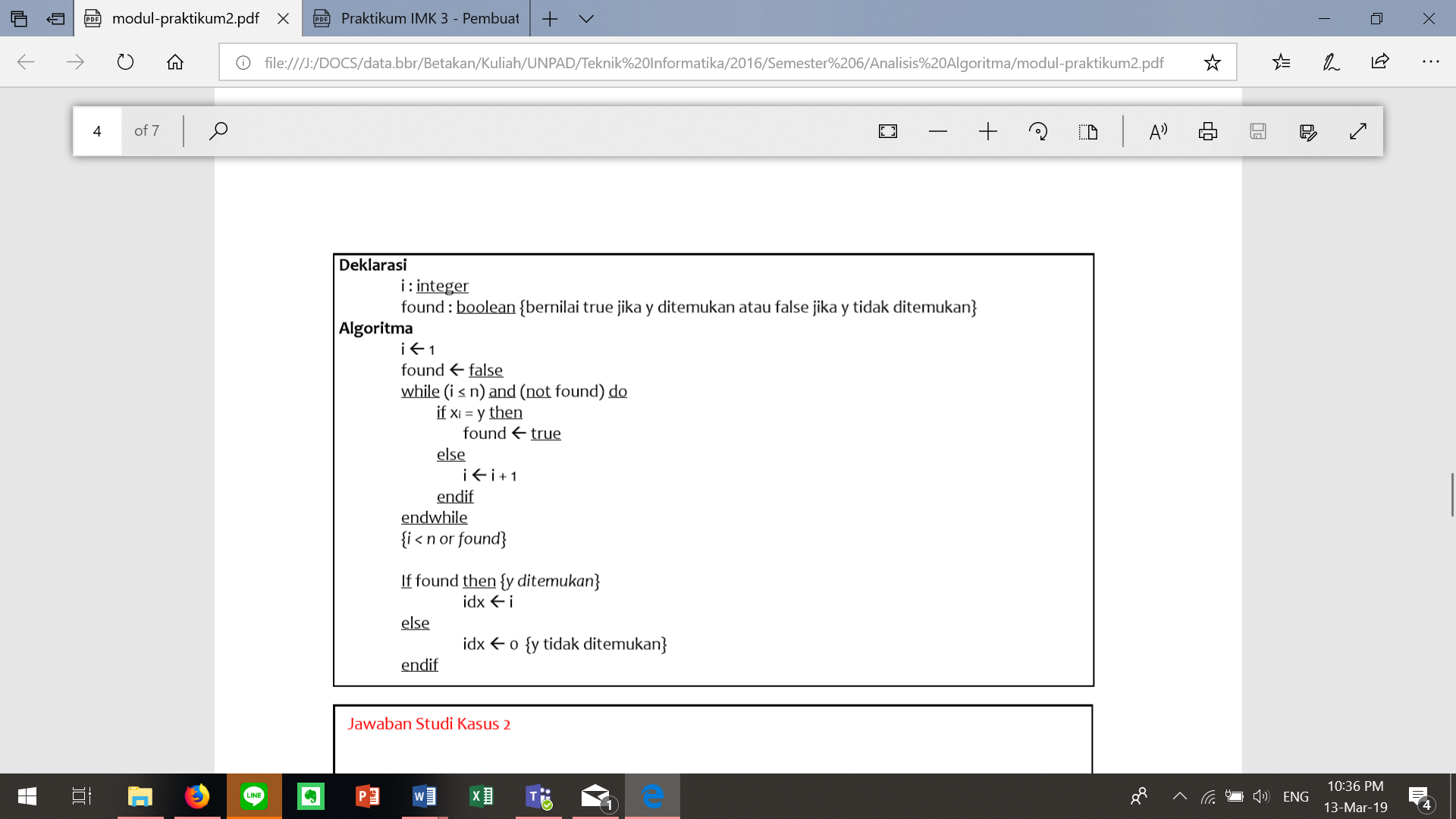
Best case: Jika nilai maks berada pada arr[0] atau index paling awal

Average case: Jika nilai maks berada pada arr[(n-1)/2] atau index di tengah

Worst case: Jika nilai maks berada pada arr[n-1] atau index paling akhir

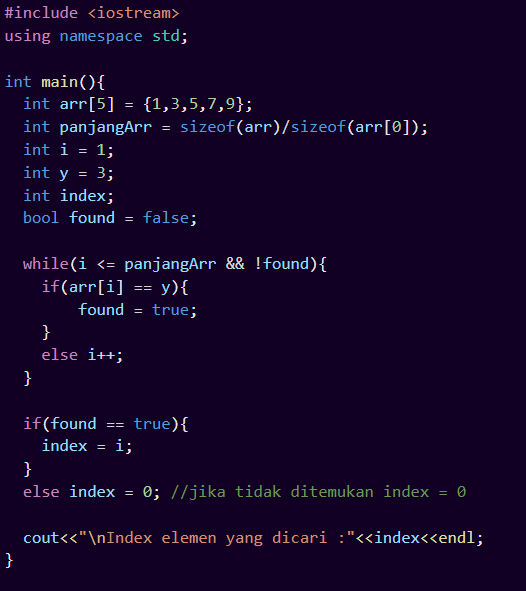
**Nomor 2**



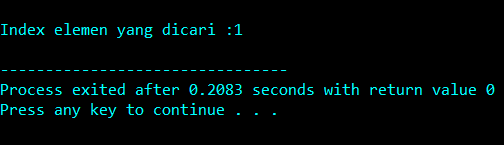


Jawab:

* Source code



* Hasil screenshoot



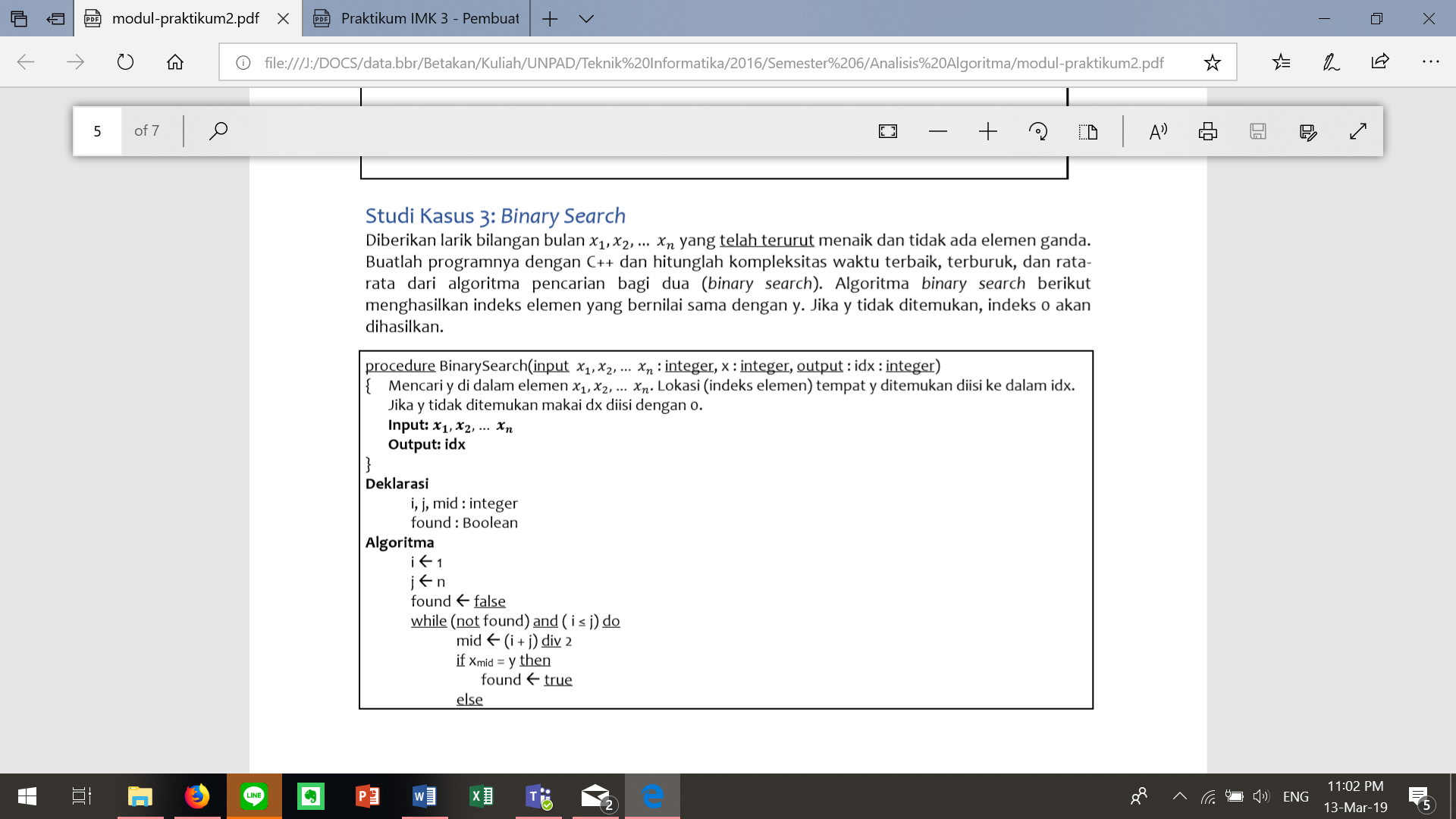
* Kompleksitas waktu:

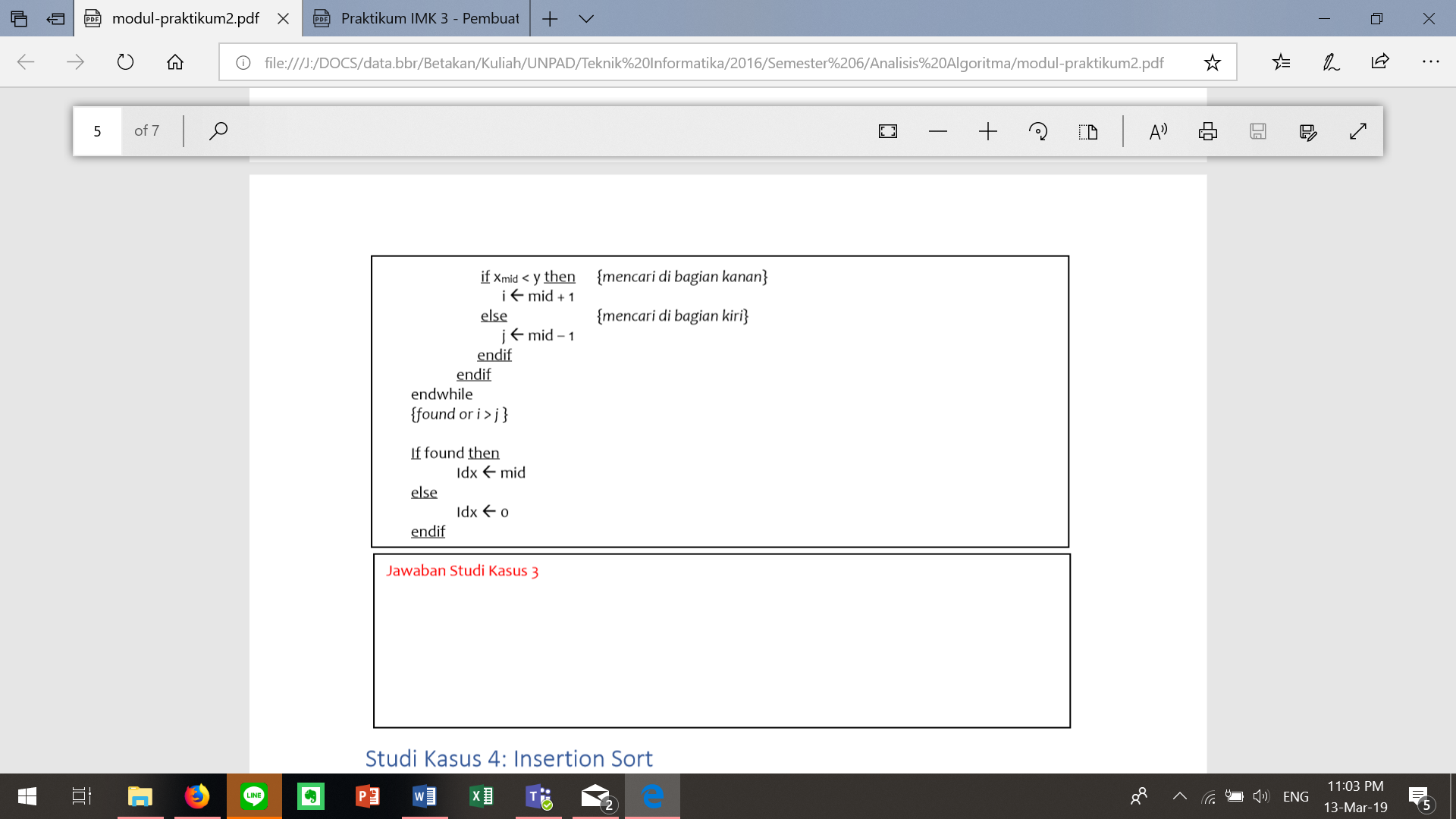
Best case: Jika ditemukan pada arr[0] atau indeks paling awal

Average case: Jika ditemukan pada arr[(n-1)/2] atau indeks di tengah

Worst case: Jika ditemukan pada arr[n-1] atau indeks paling akhir atau tidak ditemukan sama sekali

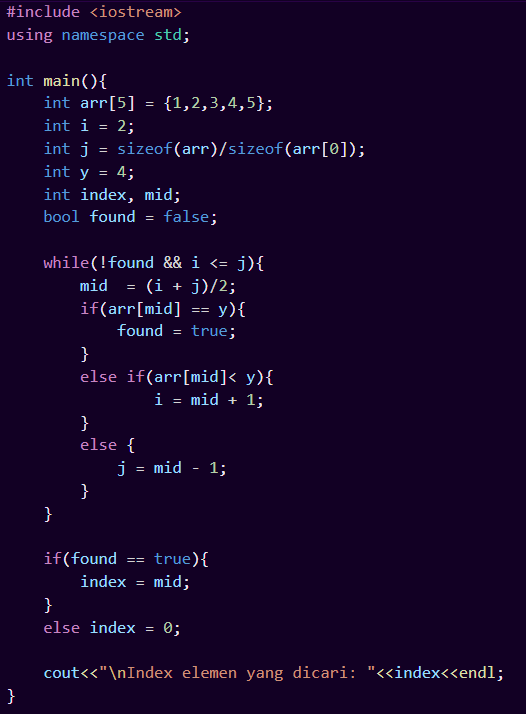
**Nomor 3**



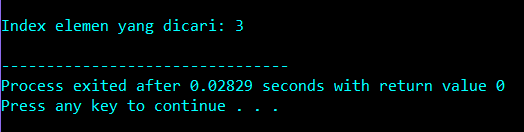


Jawab:

* Source code



* Hasil screenshoot



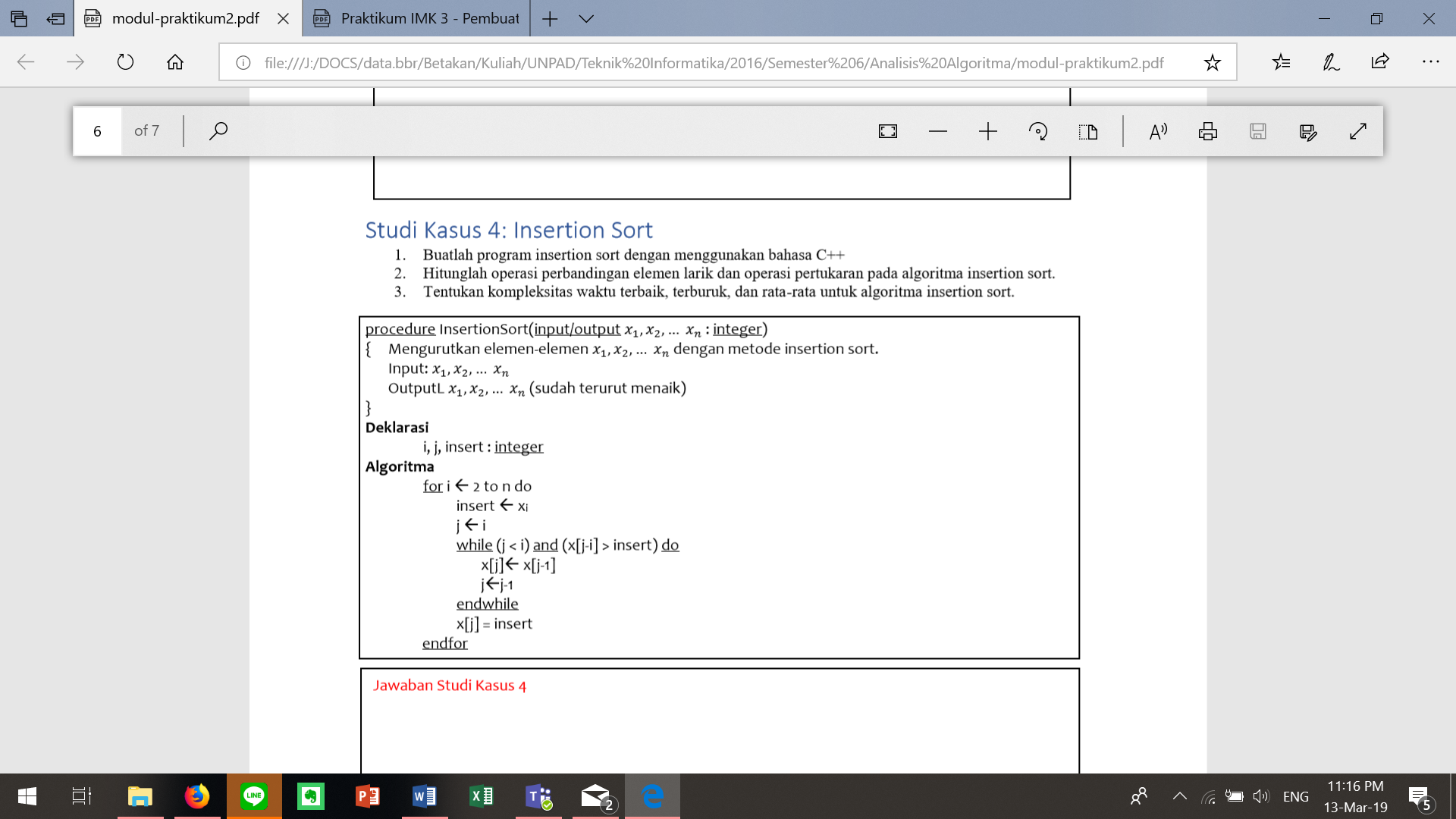
* Kompleksitas waktu:

Best case: Jika ditemukan pada arr[mid] atau indeks di tengah

Average case: Jika ditemukan pada indeks di awal atau di akhir

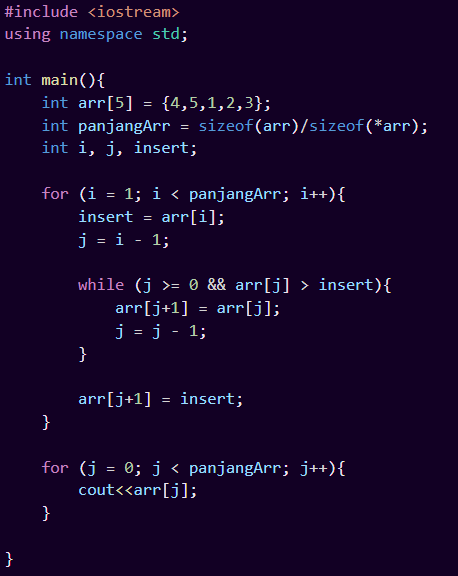
Worst case: Jika tidak ditemukan sama sekali

**Nomor 4**

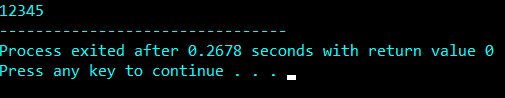


Jawab:

* Source code



* Hasil screenshot



* Kompleksitas waktu:

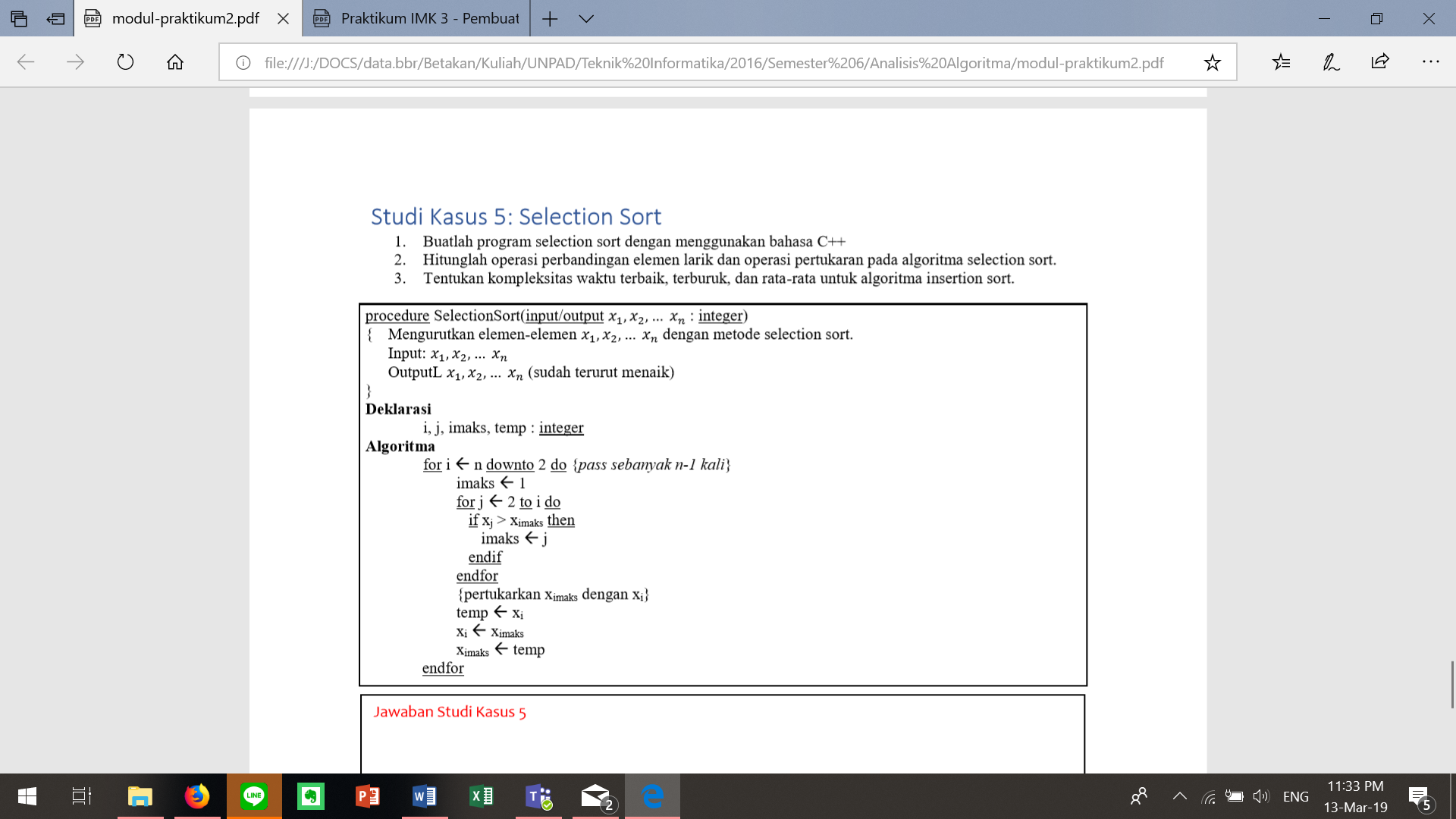
Best case: Jika array sudah terurut sehingga loop while tidak dijalankan

Average case: Jika sebagian elemen array sudah terurut

Worst case: Jika array harus diurutkan sebanyak n kali

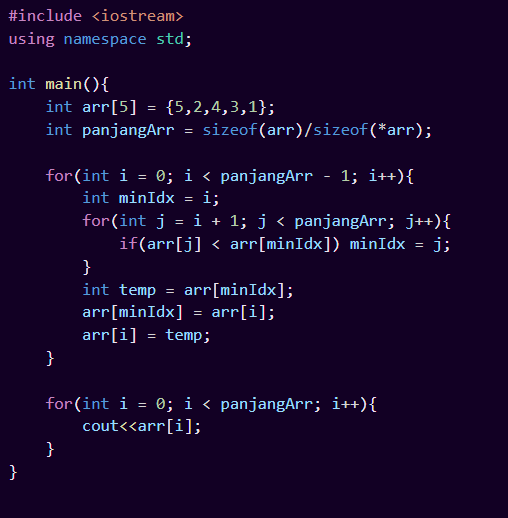
|  |  |  |  |
| --- | --- | --- | --- |
| J | Perbandingan | Perpindahan | Total operasi |
| 2 | 1 | 1 | 2 |
| 3 | 2 | 2 | 4 |
| 4 | 3 | 3 | 6 |
| n | (n-1) | (n-1) | **2(n-1)** |

**Nomor 5**

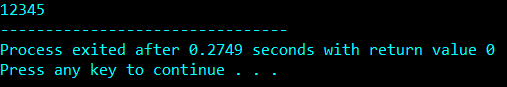


Jawab:

* Source code



* Hasil screenshoot



* Kompleksitas waktu:

1. Jumlah operasi perbandingan elemen

Untuk setiap loop ke-i,

i = 1 🡪 jumlah perbandingan = n-1

i = 2 🡪 jumlah perbandingan = n-2

i = k 🡪 jumlah perbandingan = n-k

i = n-1 🡪 jumlah perbandingan = 1

sehingga T(n) = (n-1) + (n-2) + … + 1 = n(n-1)/2 dimana kompleksitas waktu ini berlaku menjadi yang terbaik, rata-rata maupun yang terburuk karena algoritma ini tidak melihat apakah arraynya sudah urut atau tidak terlebih dahulu.

1. Jumlah operasi pertukaran

Untuk setiap loop ke-1 sampai n-1 terjadi satu kali pertukaran elemen sehingga T(n) = n-1.