**Laporan Resmi**

**Praktikum Algoritma dan Struktur Data**

**Sequence Searching and Binary Searching**



**Dr. Tita Karlita S.Kom, M.Kom**

**Nama : Marits Ikmal Yasin**

**Kelas : 1D4 IT B**

**NRP : 3121600047**

1. Membuat menu untuk membandingkan kecepatan antara sequence search dengan binary search.

Kode :

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <sys/time.h>

#include <unistd.h>

#include <sysinfoapi.h>

#define MAKS 100000

void urutkan(int[], int);

void tampilkan(int[], int);

void sequence(int[], int, int);

void binary(int[], int, int);

void isi(int[], int[], int);

void menu();

// long gettime();

int main()

{

int panjang, pilihan, key, urut = 0;

char jwb;

// clock\_t start, end;

struct timeval stop, start;

printf("Masukkan Banyak Data : ");

scanf("%d", &panjang);

int array[panjang], sort[panjang];

isi(array, sort, panjang);

tampilkan(array, panjang);

do

{

fflush(stdin);

menu();

printf("Masukkan pilihan Anda : ");

scanf("%d", &pilihan);

if (pilihan != 1 && urut == 0)

{

urut++;

urutkan(sort, panjang);

}

printf("Masukkan data yang ingin dicari : ");

scanf("%d", &key);

gettimeofday(&start, NULL);

// long start = gettime();

// // start = clock();

// // long long int begin = GetTickCount64();

// sleep(0.10);

switch (pilihan)

{

case 1:

sequence(array, panjang, key);

break;

case 2:

sequence(sort, panjang, key);

break;

case 3:

binary(sort, panjang, key);

break;

case 4:

exit(0);

default:

puts("Input Yang Benar");

}

gettimeofday(&stop, NULL);

printf("took %lu us\n", ((stop.tv\_sec - start.tv\_sec) \* 1000000 + stop.tv\_usec - start.tv\_usec));

// end = clock();

// double extime = (double)(end - start) \* 1000 / CLOCKS\_PER\_SEC;

// printf("Searching took %f miliseconds to execute \n", extime);

// long long int end = GetTickCount64();

// double elapsed = (end - begin) \* 1e-3;

// printf("Time measured: %.3f seconds.\n", elapsed);

// long end = gettime();

// printf("Time Measured : %ld\n", end - start);

fflush(stdin);

printf("Mau lagi (y/t)? ");

scanf("%c", &jwb);

} while (jwb == 'y' || jwb == 'Y');

// total\_t = (double)(end\_t - start\_t) / CLOCKS\_PER\_SEC;

// printf("Total time taken by CPU: %f\n", total\_t);

return 0;

}

void menu()

{

puts("1. Sequential Search Unsorted Array");

puts("2. Sequential Search Sorted Array");

puts("3. Binary Search Sorted Array");

puts("4. Keluar");

}

void isi(int array[], int sort[], int panjang)

{

for (int i = 0; i < panjang; i++)

{

array[i] = sort[i] = rand();

}

}

void tampilkan(int array[], int panjang)

{

for (int i = 0; i < panjang; i++)

{

printf("%d ", array[i]);

}

puts(" ");

}

void sequence(int array[], int panjang, int key)

{

int valid = 0, perbandingan = 0;

for (int i = 0; i < panjang; i++)

{

usleep(1000);

perbandingan++;

if (key == array[i])

{

printf("%d ada di dalam Array indeks ke %d\n", key, i);

valid = 1;

break;

}

}

if (!valid)

{

puts("Data tidak ditemukan di dalam Array");

}

printf("Perbandingan : %d\n", perbandingan);

}

void binary(int array[], int panjang, int key)

{

int kiri = 0, kanan = panjang - 1, tengah, valid = 0, perbandingan = 0;

while (kiri <= kanan)

{

usleep(1000);

perbandingan++;

tengah = (kiri + kanan) / 2;

if (array[tengah] == key)

{

printf("%d ada di dalam Array indeks ke %d\n", key, tengah);

valid = 1;

break;

}

else if (array[tengah] < key)

{

kiri = tengah + 1;

}

else

{

kanan = tengah - 1;

}

}

if (!valid)

{

puts("Data tidak ditemukan di dalam Array");

}

printf("Perbandingan : %d\n", perbandingan);

}

// bubble sort

void urutkan(int sort[], int panjang)

{

int cek, akhir = panjang, temp;

for (int i = 0; i < panjang; i++)

{

cek = 0;

akhir--;

for (int j = 0; j < akhir; j++)

{

if (sort[j] > sort[j + 1])

{

temp = sort[j];

sort[j] = sort[j + 1];

sort[j + 1] = temp;

}

else

{

cek++;

if (cek == panjang - 1)

puts("Data telah diurutkan");

}

}

}

tampilkan(sort, panjang);

}

// long gettime()

// {

// struct timeval start, end;

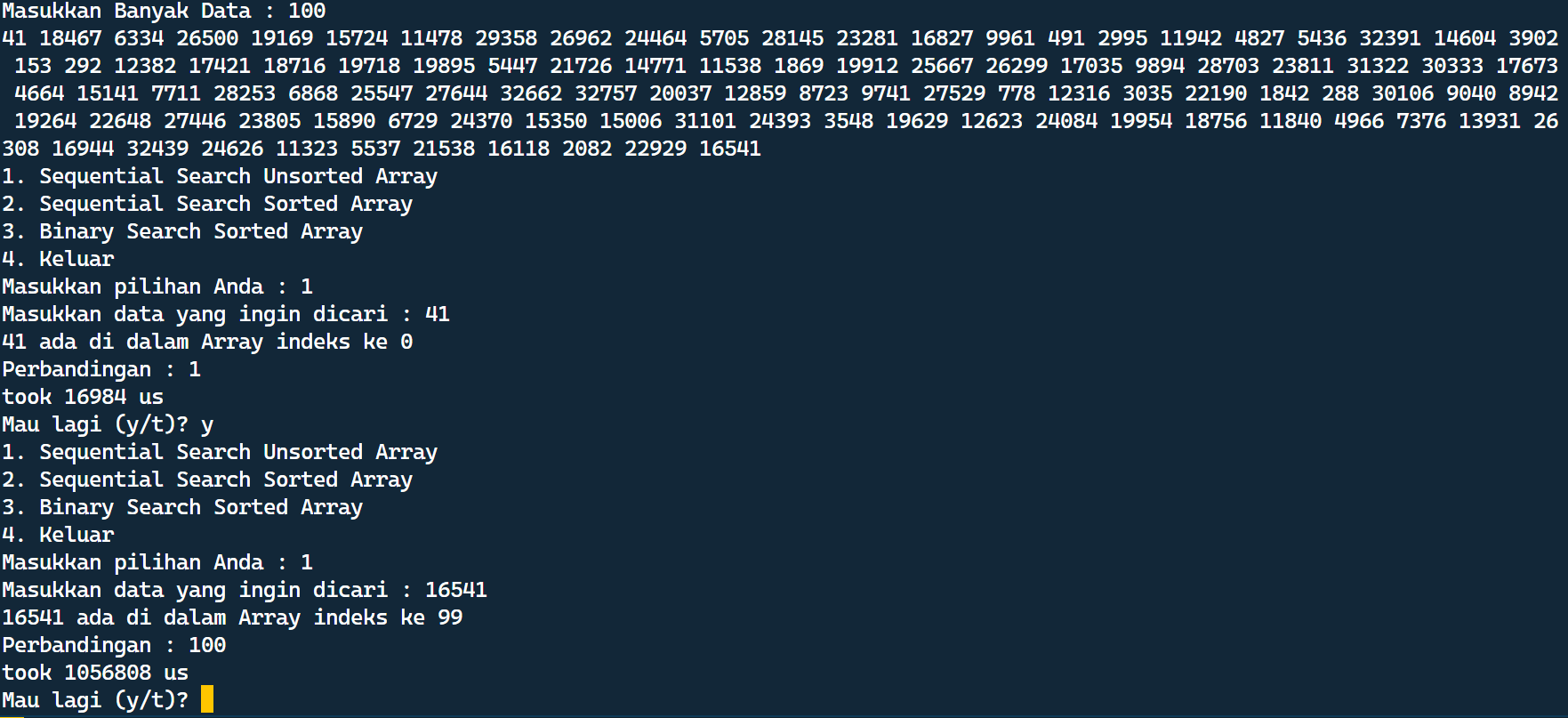
// gettimeofday(&start, NULL);

// return start.tv\_sec \* 1000 + start.tv\_usec;

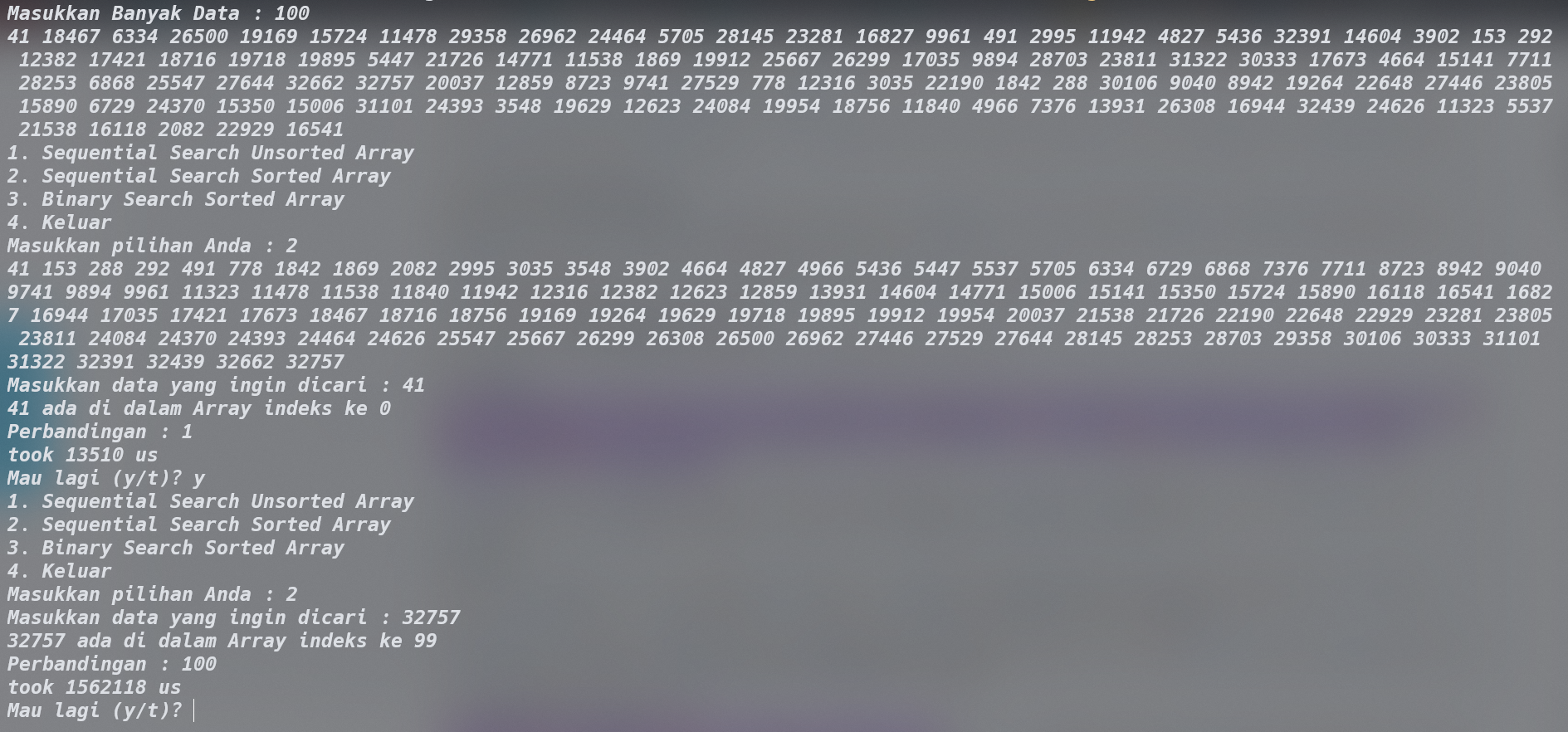
// }

Output :

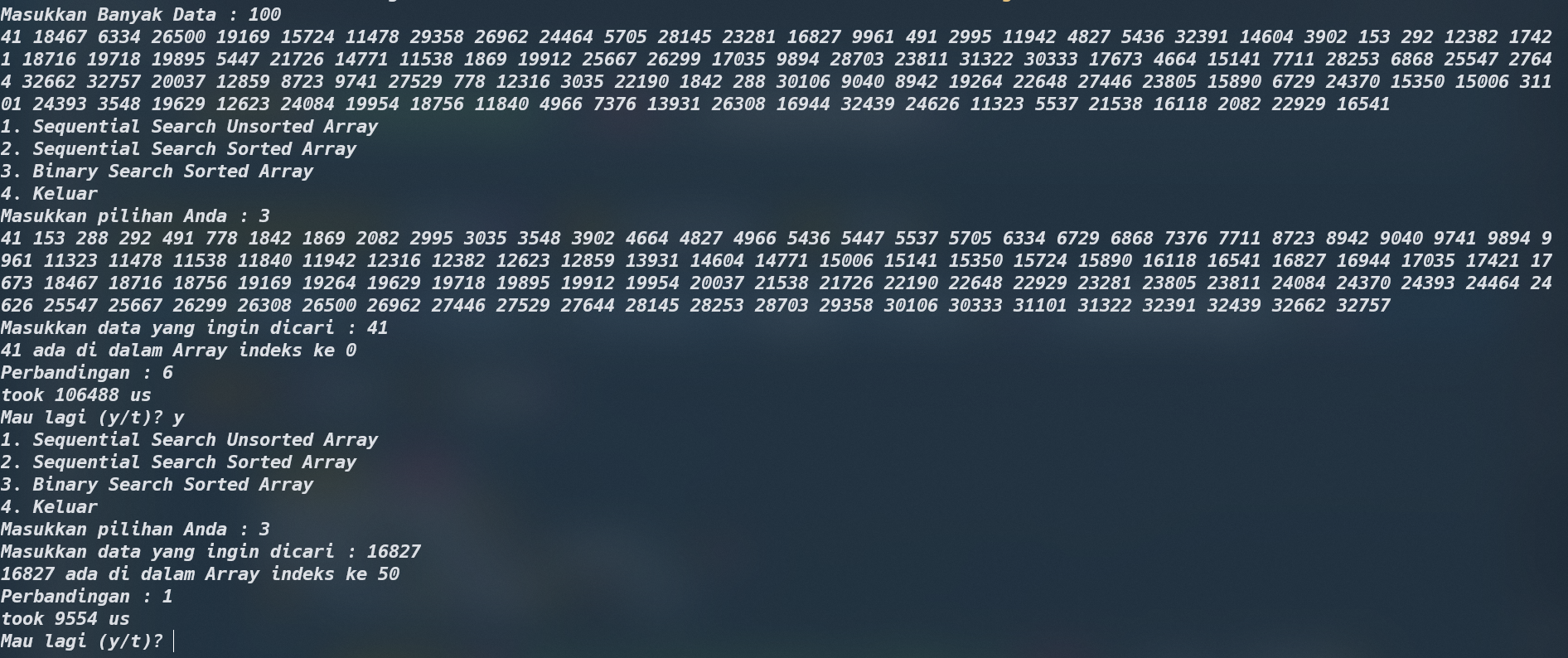
Pilihan 1 dengan memilih indeks 0 dan 99.



Pilihan 2 dengan memilih indeks 0 dan 99.



Pilihan 3 dengan memilih indeks 0 dan 99.



**Analisa :**

Waktu yang diperlukan dalam menggunakan sequence search cenderung lebih lama. Sedangkan saat kita menggunakan binary search lebih cepat. Hal itu disebabkan karena saat menggunakan sequence search kita perlu mengecek data satu-persatu sesuai urutan array. Sedangkan binary search melakukan perbandingan dengan data yang berada indeks di tengah. Oleh karena itu binary search lebih cepat daripada sequence search.