**Laporan Resmi**

**Praktikum Algoritma dan Struktur Data**

**Insertion and Selection Sort**



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

**Nama : Marits Ikmal Yasin**

**Kelas : 1D4 IT B**

**NRP : 3121600047**

1. Kode yang menampilkan waktu eksekusi, jml perbandingan dan jml pertukaran untuk seribu data random untuk selection dan insertion sort
2. Selection Sort

Kode :

#include <stdio.h>

#include <stdlib.h>

#include <sys/time.h>

#include <unistd.h>

#define MAKS 1000

int perbandingan, penukaran;

void printArray(int arr[], int sz)

{

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

{

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

}

printf("\n");

}

void selectionSort(int arr[], int sz)

{

for (int step = 0; step < sz - 1; step++)

{

int min = step;

for (int i = step + 1; i < sz; i++)

{

usleep(100);

perbandingan++;

if (arr[i] < arr[min])

{

min = i;

}

}

if (arr[min] != arr[step])

penukaran++;

int temp = arr[min];

arr[min] = arr[step];

arr[step] = temp;

}

}

int main()

{

struct timeval start, stop;

int array[MAKS];

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

{

array[i] = rand() % 1000;

}

int size = sizeof(array) / sizeof(array[0]);

puts("Data Sebelum Sorting");

printArray(array, size);

gettimeofday(&start, NULL);

selectionSort(array, size);

gettimeofday(&stop, NULL);

puts("Data Setelah Sorting");

printArray(array, size);

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

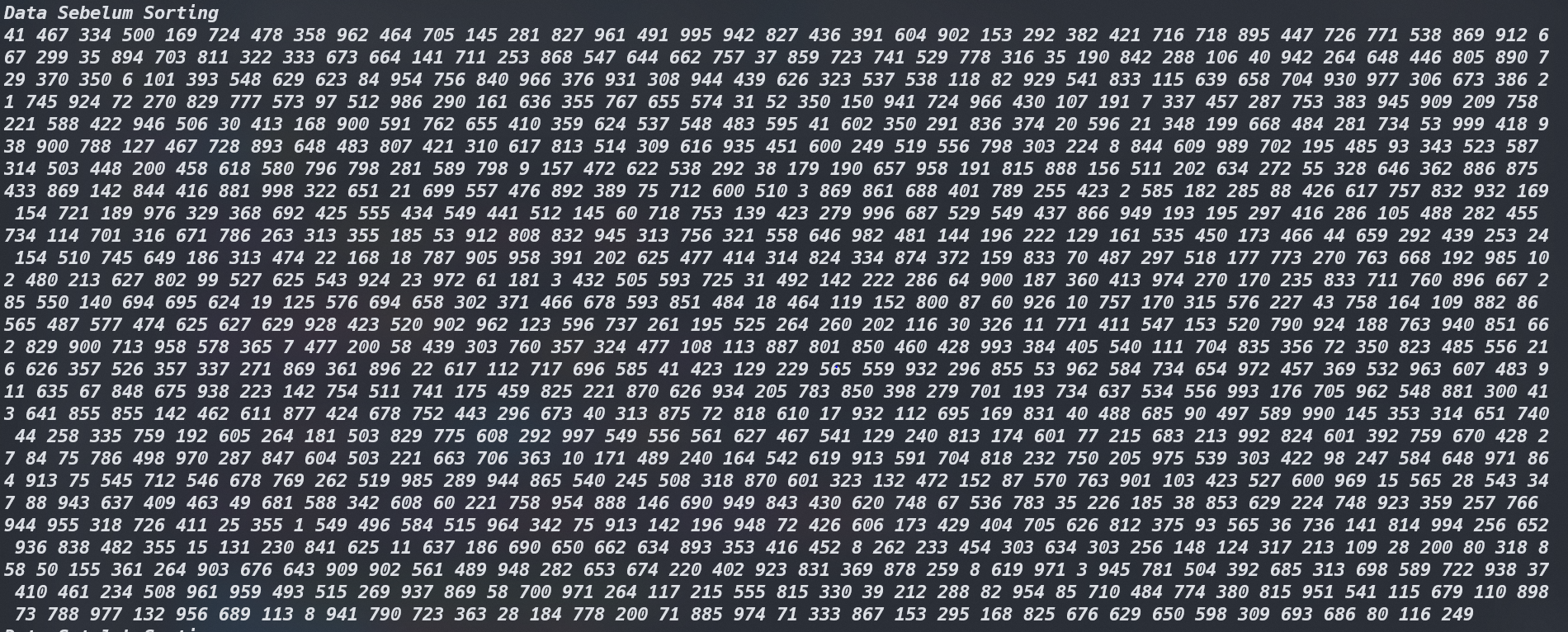
printf("Perbandingan = %d\n", perbandingan);

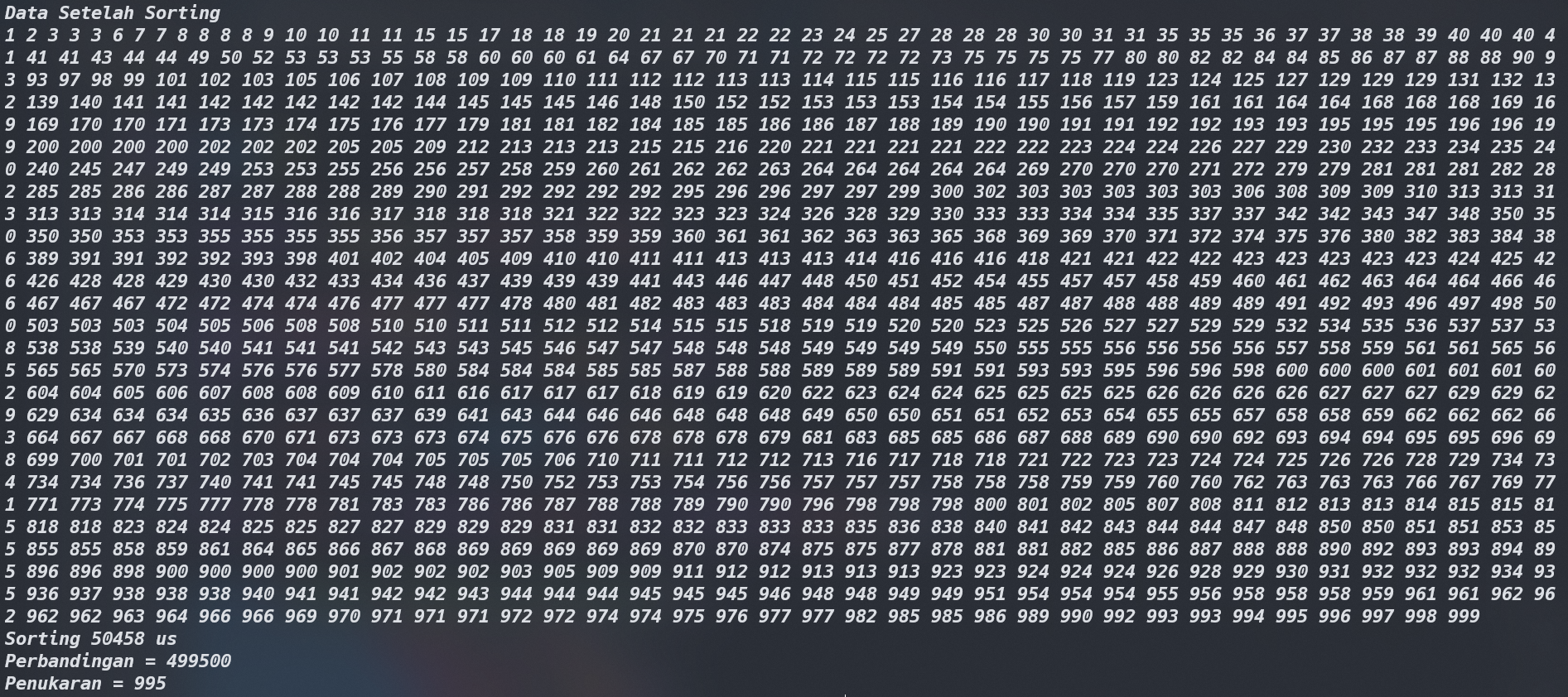
printf("Penukaran = %d\n", penukaran);

return 0;

}

Output :





1. Insertion Sort

Kode :

#include <stdio.h>

#include <stdlib.h>

#include <sys/time.h>

#include <unistd.h>

#define MAKS 1000

int perbandingan, pergeseran, penyisipan;

void printArray(int arr[], int sz)

{

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

{

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

}

printf("\n");

}

void insertionSort(int arr[], int sz)

{

for (int step = 1; step < sz; step++)

{

int key = arr[step];

int back = step - 1;

// best case comparison is n-1

while (key < arr[back] && back >= 0)

{

// worst case comparison is n(n-1)/2

usleep(100);

pergeseran++;

arr[back + 1] = arr[back];

back--;

perbandingan++;

}

if (key > arr[back])

{

perbandingan++;

usleep(100);

}

if (arr[back + 1] != key)

penyisipan++;

arr[back + 1] = key;

}

}

int main()

{

struct timeval start, stop;

int array[MAKS];

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

{

array[i] = rand();

}

int size = sizeof(array) / sizeof(array[0]);

puts("Data Sebelum Sorting");

printArray(array, size);

gettimeofday(&start, NULL);

insertionSort(array, size);

gettimeofday(&stop, NULL);

puts("Data Setelah Sorting");

printArray(array, size);

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

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

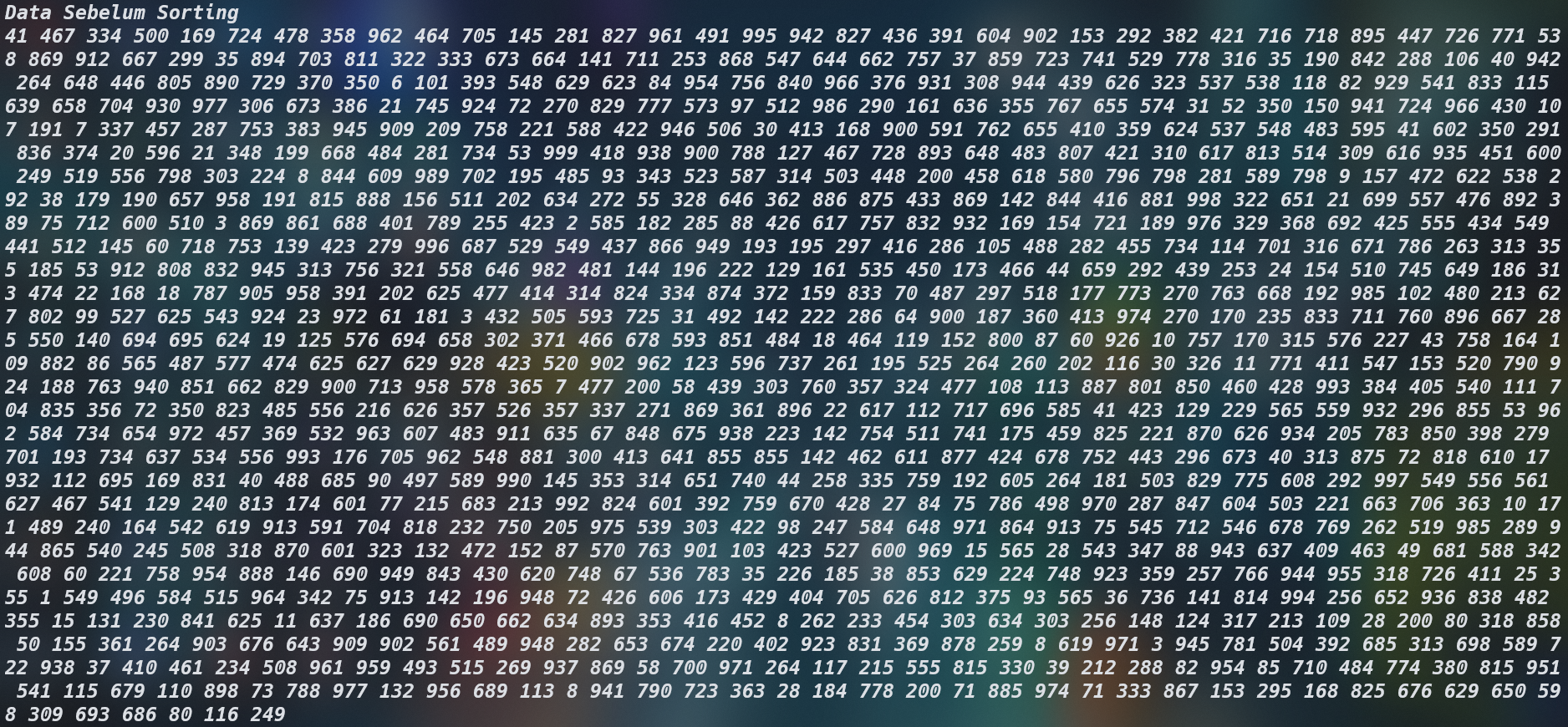
printf("Pergeseran = %d\n", pergeseran);

printf("Penyisipan = %d\n", penyisipan);

return 0;

}

Output :





1. Data Struct

Kode :

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

#define MAKS 25

typedef struct

{

int no;

char nama[MAKS];

int nilai;

} siswa;

int banyak;

void menu()

{

puts("MENU METODE SORTING");

puts("1. Insertion Sort");

puts("2. Selection Sort");

puts("3. Keluar");

}

void selection()

{

puts("MENU METODE SORTING");

puts("1. Ascending");

puts("2. Descending");

}

void insertion()

{

puts("Pengurutan yang dipilih :");

puts("1. Ascending");

puts("2. Descending");

}

void input(siswa arr[], int sz)

{

char jwb;

static int i = 0, cek = 0;

if (cek != 0)

{

printf("Mau Input Data Lagi (y/t)? ");

fflush(stdin);

jwb = getchar();

fflush(stdin);

toupper(jwb);

}

if (cek == 0 || jwb == 'y')

{

cek++;

do

{

fflush(stdin);

printf("Masukkan data ke %d\n", ++i);

printf("Masukkan No : ");

scanf("%d", &arr[banyak].no);

fflush(stdin);

printf("Masukkan Nama : ");

gets(arr[banyak].nama);

fflush(stdin);

printf("Isi nilai : ");

scanf("%d", &arr[banyak].nilai);

fflush(stdin);

banyak++;

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

jwb = getchar();

puts(" ");

jwb = toupper(jwb);

if (jwb == 'Y' && banyak == 9)

{

puts("Tidak Bisa Input lagi !");

exit(0);

}

} while (jwb == 'Y' && banyak < sz);

}

}

// Ascending

void AscSelectionSort(siswa arr[])

{

for (int step = 0; step < banyak - 1; step++)

{

int min = step;

for (int i = step + 1; i < banyak; i++)

{

if (arr[i].no < arr[min].no)

min = i;

}

siswa temp = arr[min];

arr[min] = arr[step];

arr[step] = temp;

}

}

// Descending

void DescSelectionSort(siswa arr[])

{

for (int step = 0; step < banyak - 1; step++)

{

int min = step;

for (int i = step + 1; i < banyak; i++)

{

if (arr[i].no > arr[min].no)

min = i;

}

siswa temp = arr[min];

arr[min] = arr[step];

arr[step] = temp;

}

}

// Ascending Insertion

void AscInsertionSort(siswa arr[])

{

for (int step = 1; step < banyak; step++)

{

siswa temp = arr[step];

int key = arr[step].no;

int back = step - 1;

while (key < arr[back].no && back >= 0)

{

arr[back + 1] = arr[back];

back--;

}

arr[back + 1] = temp;

}

}

// Descending Insertion

void DescInsertionSort(siswa arr[])

{

for (int step = 1; step < banyak; step++)

{

siswa temp = arr[step];

int key = arr[step].no;

int back = step - 1;

while (key > arr[back].no && back >= 0)

{

arr[back + 1] = arr[back];

back--;

}

arr[back + 1] = temp;

}

}

void printArray(siswa arr[])

{

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

{

printf("No\t: %d\n", arr[i].no);

printf("Nama\t: %s\n", arr[i].nama);

printf("Nilai\t: %d\n", arr[i].nilai);

puts(" ");

}

puts(" ");

}

void copyData(siswa Data[], siswa Data2[])

{

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

{

Data2[i] = Data[i];

}

}

int main()

{

int pilihan, pilih;

siswa Data[10], Data2[10];

int size = sizeof(Data) / sizeof(Data[0]);

char jawab;

do

{

fflush(stdin);

input(Data, size);

copyData(Data, Data2);

printArray(Data2);

menu();

printf("Pilihan Anda[1/2/3] : ");

scanf("%d", &pilihan);

switch (pilihan)

{

case 1:

insertion();

break;

case 2:

selection();

break;

default:

exit(0);

}

printf("Pilihan Anda[1/2] : ");

scanf("%d", &pilih);

if (pilihan == 1)

{

switch (pilih)

{

case 1:

AscInsertionSort(Data2);

break;

case 2:

DescInsertionSort(Data2);

break;

default:

exit(0);

}

}

else

{

switch (pilih)

{

case 1:

AscSelectionSort(Data2);

break;

case 2:

DescInsertionSort(Data2);

break;

default:

exit(0);

}

}

printArray(Data2);

fflush(stdin);

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

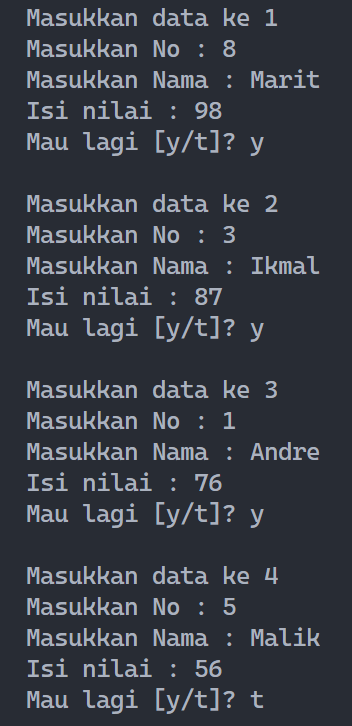
jawab = getchar();

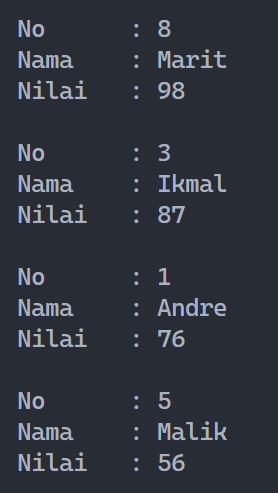
jawab = toupper(jawab);

} while (jawab == 'Y');

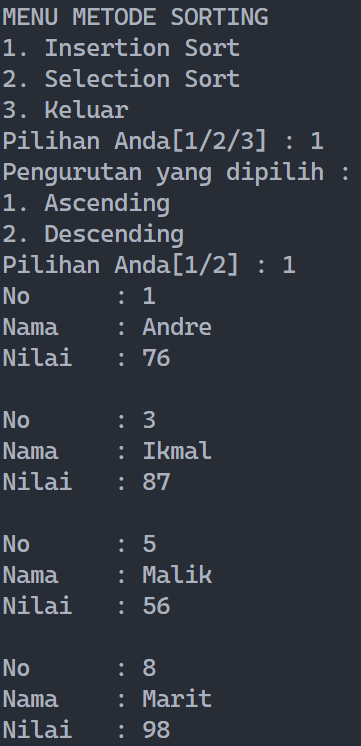
}

Output :

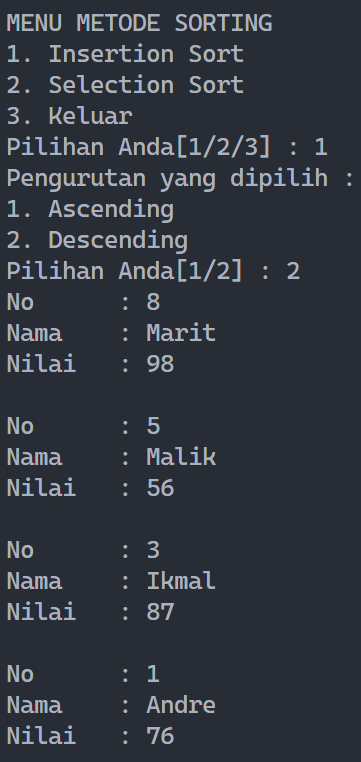




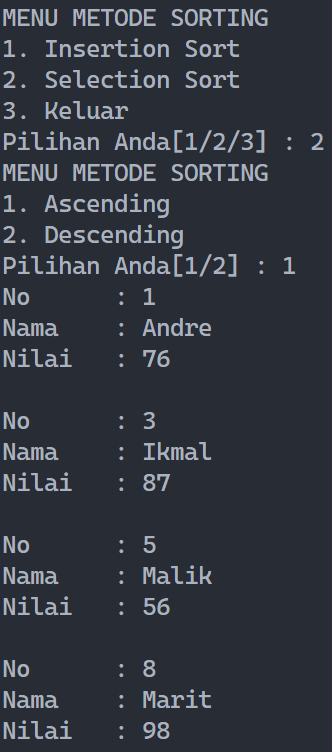
1. Insertion Ascending



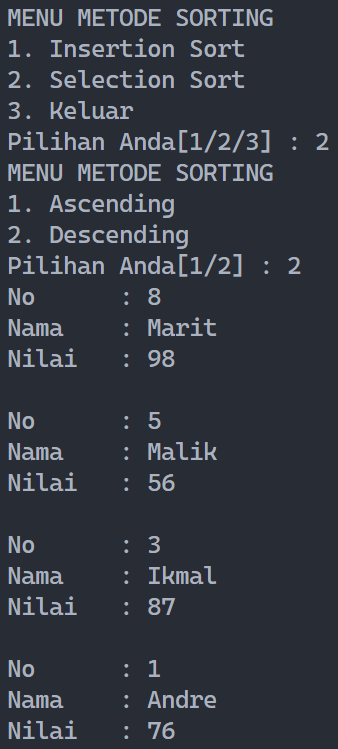
1. Insertion Descending



1. Selection Ascending



1. Selection Descending



Kesimpulan :

Waktu kompleksitas pada Selection sort pada semua kasus yaitu O(n2). Sedangkan Insertion sort antara best case dan worst casenya berbeda. Insertion sort best case memiliki waktu kompleksitas O(n) dan untuk worst casenya O(n2). Jadi apabila membicarakan mana yang lebih cepat antara Selection sort dan Insertion sort maka insertion akan lebih cepat karena lebih stabil. Semakin sedikit perbandingan yang dilakukan maka semakin cepat pula proses pengurutan. Beda halnya dengan selection sort. Apabila data banyak mau sudah terurut atau belum maka kecepatannya sama. Hal ini didasari oleh waktu komplesitas di atas.