Tugas 2

Praktikum Analisis Algoritma



Muhammad Naufal Monoarfa 140810160033

S-1 Teknik Informatika

Fakultas Matematika & Ilmu Pengetahuan Alam

Universitas Padjadjaran

1. Program:

#include <iostream>

using namespace std;

main() {

  int n;

  int x[10];

  cout<< " Jumlah Data : ";

  cin>> n;

  for (int i = 0; i < n; i++){

    cout<< "Data ke - " << i+1 << " : ";

    cin>> x[i];

  }

  int maks = x[0];

  int i = 1;

  while (i <= n){

    if (x[i] > maks)

      maks = x[i];

    i++;

  }

  cout << "Nilai Maks : " << maks << endl;

}

Kompleksitas:

maks 🡨 x1 1 kali

i 🡨 2 1 kali

maks 🡨 xi  n kali

i 🡨 i + 1 n kali

1. Program

#include <iostream>

using namespace std;

main() {

  int n;

  int x[10];

  cout << "Jumlah Data : ";

  cin >> n;

  for (int i = 0; i < n; i++){

    cout << "Masukkan Data ke - " << i+1 << " : ";

    cin >> x[i];

  }

  int y;

  cout << "Masukkan key : ";

  cin >> y;

  int i = 0;

  bool found = false;

  int idx;

  while ((i < n) && (!found)){

    if (x[i] == y)

      found = true;

    else

      i++;

  }

  if (found)

    idx = i+1;

  else

    idx = 0;

  cout << "Yang dicari berada di urutan : " << idx << endl;

}

## Kompleksitas:

Best:

i 🡨 1 1 kali

found false 1 kali

found true 1 kali

idx 🡨 I 1 kali

Average:

i 🡨 1 1 kali

found false 1 kali

i 🡨 i + 1 ½ n kali

found true 1 kali

idx🡨 I 1 kali

Worst:

i 🡨 1 1 kali

found🡨 false 1 kali

i 🡨 i + 1 n kali

found true 1 kali

idx🡨 I 1 kali

1. Program:

#include <iostream>

using namespace std;

main() {

  int n;

  int x[10];

  cout << "Jumlah Data : ";

  cin >> n;

  for (int i = 0; i < n; i++){

    cout << "Data ke - " << i+1 << " : ";

    cin >> x[i];

  }

  int y;

  cout << "Masukkan yang dicari : ";

  cin >> y;

  int i = 0;

  int j = n-1;

  bool found = false;

  int idx;

  int mid;

  while ((i <= j) && (!found)){

    mid = (i + j)/2;

    if (x[mid] == y)

      found = true;

    else{

      if (x[mid] < y)

        i = mid + 1;

      else

        j = mid - 1;

    }

  }

  if (found)

    idx = mid+1;

  else

    idx = 0;

  cout << "Yang dicari berada di index ke : " << idx << endl;

}

Kompleksitas

Best:

i 🡨 1 1 kali

j 🡨 n 1 kali

found 🡨 false 1 kali

mid 🡨 (i + j) div2 1 kali

found 🡨 true 1 kali

Idx 🡨 mid 1 kali

Average:

i 🡨 1 1 kali

j 🡨 n 1 kali

found 🡨 false 1 kali

mid 🡨 (i + j) div2 ½ n + 1 kali

i 🡨 mid + 1 or j 🡨 mid –1 ½ n kali

found 🡨 true 1 kali

Idx 🡨 mid 1 kali

`

Worst

i 🡨 1 1 kali

j 🡨 n 1 kali

found 🡨 false 1 kali

mid 🡨 (i + j) div2 n + 1 kali

i 🡨 mid + 1 or j mid –1 n kali

found true 1 kali

Idx mid 1 kali

1. Program:

#include <iostream>

using namespace std;

main(){

  int n;

  int x[10];

  cout << "Jumlah Data : ";

  cin >> n;

  for (int i = 0; i < n; i++){

    cout << "Data ke - " << i+1 << " : ";

    cin >> x[i];

  }

  cout << "Data Sebelum di Sorting : ";

  for (int i = 0; i < n; i++)

    cout << x[i] << " ";

  cout << endl;

  int insert;

  int j;

  for (int i = 1; i < n; i++){

    insert = x[i];

    j = i-1;

    while ((j >= 0) && (x[j] > insert)){

      x[j+1] = x[j];

      j--;

    }

    x[j+1] = insert;

  }

  cout << "Setelah di Sorting : ";

  for (int i = 0; i < n; i++)

    cout << x[i] << " ";

}

### Kompleksitas

### Best

For i 🡨 2 to n do 1 kali

insert 🡨 xi n kali

j 🡨 i n kali

x[j] = insert n kali

Avarage

For i 🡨 2 to n do 1 kali

insert 🡨 xi n kali

j 🡨 I n kali

x[j] 🡨 x[j-1] n \* ½ n kali

j🡨 j-1 n \* ½ n kali

x[j] = insert n kali

Worst

For i 🡨 2 to n do 1 kali

insert 🡨 xi n kali

j 🡨 i n kali

x[j] 🡨 x[j-1] n \* n kali

j🡨 j-1 n \* n kali

x[j] = insert n kali

1. Program:

#include <iostream>

using namespace std;

main(){

  int n;

  int x[10];

  cout << "Jumlah Data : ";

  cin >> n;

  for (int i = 0; i < n; i++){

    cout << "Data ke - " << i+1 << " : ";

    cin >> x[i];

  }

  cout << "Sebelum di Sorting : ";

  for (int i = 0; i < n; i++)

    cout << x[i] << " ";

  cout << endl;

  int imaks;

  int temp;

  for (int i = n-1; i >= 1; i--){

    imaks = 0;

    for (int j = 1; j <= i; j++){

      if (x[j] > x[imaks])

        imaks = j;

    }

    temp = x[i];

    x[i] = x[imaks];

    x[imaks] = temp;

  }

  cout << "Data setelah di Sorting : ";

  for (int i = 0; i < n; i++)

    cout << x[i] << " ";

}

Kompleksitas:

### Best

for i 🡨 n downto 2 do 1 kali

imaks 🡨 1 n kali

for j 🡨 2 to i do n kali

imaks 🡨 j n\*1 kali

temp 🡨 xi n kali

xi🡨 ximaks n kali

ximaks🡨 temp n kali

Avarage

for i 🡨 n downto 2 do 1 kali

imaks 🡨 1 nkali

for j 🡨 2 to i do n kali

imaks 🡨 j n \* ½ n kali

temp 🡨 xi n kali

xi🡨 ximaks n kali

ximaks🡨 temp n kali

Worst

for i 🡨 n downto 2 do 1 kali

imaks 🡨 1 n kali

for j 🡨 2 to i do n kali

imaks 🡨 j n \* n kali

temp 🡨 xi n kali

xi🡨 ximaks n kali

ximaks🡨 temp n kali