**Tugas 3**

**Analisis Algoritma**



Disusun oleh :

Afifah Kho’eriah (140810160008)

Baby Cattleya Gustina Permatagama (140810160048)

Muhammad Islam Taufikurahman (140810160062)

S-1 Teknik Informatika   
Fakultas Matematika & Ilmu Pengetahuan Alam   
Universitas Padjadjaran  
Jalan Raya Bandung - Sumedang Km. 21 Jatinangor 45363

**Program Counting Sort**

|  |
| --- |
| **// // Nama: Program Counting Sort (C++) // Kelompok: Afifah 140810160008, Baby 140810160048, Muhammad Islam 140810160062 // Mata Kuliah: Analisis Algoritma // // #include <iostream> #include <chrono> #include <ctime> #include <cstdlib>  using namespace std;  int k = 0;  // Method untuk melakukan sorting pada Array void Counting\_Sort(int A[], int B[], int n) {    int C[k];  for(int i = 0; i < k+1; i++) {  // Inisiasi array C == 0  C[i]=0;  }   for(int j = 1; j <= n; j++) {  // Menghitung kemunculan setiap elemen x dalam A  // dan menambahkannya pada posisi x di C  C[A[j]]++;   }   for(int i = 1; i <= k; i++) {  // Menyimpan kemunculan elemen i terakhir  C[i] += C[i-1];  }   for(int j = n; j >= 1; j--) {  // Menaruh elemen pada tempatnya  B[C[A[j]]] = A[j];   // elemen yang muncul dua kali akan membuat lebih mudah  C[A[j]] = C[A[j]]-1;   } }  int main() {  int n;  cout << "Masukan panjang array :";  cin >> n;    cout << "Data array sebelum sort :";    /\*A, menyimpan elemen yang dimasukan oleh user ke array \*/  /\*B, menyimpan hasil sorting\*/   int A[n], B[n];    unsigned seed = time(0);  srand(seed);    // Buat Random Array  for(int i = 1; i<=n; i++)  {  A[i]=rand()%10+1;  }    // Buat nilai k dan cetak Array terbentuk  for(int i = 1; i <= n; i++) {  cout << A[i] << " ";  if(A[i] > k) {  // Merubah nilai k jika pada elemen i nilainya lebih besar dari k  k = A[i];   }  }   auto start = chrono::steady\_clock::now();    Counting\_Sort(A, B, n);   auto end = chrono::steady\_clock::now();  auto diff = end - start;  // Print array yang telah di sorting    cout << endl << "Data array setelah sort :";  for(int i = 1; i <= n; i++) {  cout << B[i] << " ";  }    cout << endl <<"Runtime : " << chrono::duration <double, milli> (diff).count() << " ms" << endl;  return 0; }** |

**Gambaran Algoritma Counting Sort**

Nama array : A, B, C

n : 8

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

dan array C setelah diinisialisasikan :

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 |

1 2 3 4 5 6

Penjelasan

Langkah 1 : pembacaan pertama mendapat elemen A[1] dengan isi 3, maka C[3] ditambah 1

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 0 | 0 | 0 |

1 2 3 4 5 6

Langkah 2 : pembacaan kedua mendapat elemen A[2] dengan isi 6, maka C[6] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 0 | 0 | 1 |

1 2 3 4 5 6

Langkah 3 : pembacaan ketiga mendapat elemen A[3] dengan isi 4, maka C[4] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 1 | 0 | 1 |

1 2 3 4 5 6

Langkah 4 : pembacaan keempat mendapat elemen A[4] dengan isi 1, maka C[1] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 0 | 1 |

1 2 3 4 5 6

Langkah 5 : pembacaan kelima mendapat elemen A[5] dengan isi 3, maka C[3] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 2 | 1 | 0 | 1 |

1 2 3 4 5 6

Langkah 6 : pembacaan keenam mendapat elemen A[6] dengan isi 4, maka C[4] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 0 | 2 | 2 | 0 | 1 |

1 2 3 4 5 6

Langkah 7 : pembacaan ketujuh mendapat elemen A[7] dengan isi 1, maka C[1] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 0 | 2 | 2 | 0 | 1 |

1 2 3 4 5 6

Langkah 8 : pembacaan kedelapan mendapat elemen A[8] dengan isi 4, maka C[4] ditambah 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 0 | 2 | 3 | 0 | 1 |

1 2 3 4 5 6

Maka Array C setelah melewati 8 langkah

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 0 | 2 | 3 | 0 | 1 |

1 2 3 4 5 6

Lalu dilakukan proses penambahan pada setiap larik, sehingga C menjadi

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 2 | 4 | 7 | 7 | 8 |

1 2 3 4 5 6

Dalam proses ini kita mengakses elemen A[i], kemudian memposisikannya di posisi sebagaimana tercatat dalam C[A[i]], kemudian kita mengurangkan C[A[i]] dengan 1, yang dengan jelas untuk memberikan posisi untuk elemen berikutnya dengan yang isinya sama dengan A[i]. Kita buat array B.

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | - | - | - | - | - | - | - |

1 2 3 4 5 6 7 8

Langkah 1 : elemen A[8] adalah 4, maka karena C[4] adalah 7, maka B[7] diisi dengan 4, dan C[4] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | - | - | - | - | - | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 2 | 4 | 6 | 7 | 8 |

1 2 3 4 5 6

Langkah 2 : elemen A[7] adalah 1, maka karena C[1] adalah 2, maka B[2] diisi dengan 1, dan C[1] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | 1 | - | - | - | - | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 4 | 6 | 7 | 8 |

1 2 3 4 5 6

Langkah 3 : elemen A[6] adalah 4, maka karena C[4] adalah 6, maka B[6] diisi dengan 4, dan C[4] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | 1 | - | - | - | 4 | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 4 | 5 | 7 | 8 |

1 2 3 4 5 6

Langkah 4 : elemen A[5] adalah 3, maka karena C[3] adalah 4, maka B[4] diisi dengan 3, dan C[3] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| - | 1 | - | 3 | - | 4 | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 5 | 7 | 8 |

1 2 3 4 5 6

Langkah 5 : elemen A[4] adalah 1, maka karena C[1] adalah 1, maka B[1] diisi dengan 1, dan C[1] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | - | 3 | - | 4 | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 2 | 3 | 5 | 7 | 8 |

1 2 3 4 5 6

Langkah 6 : elemen A[3] adalah 4, maka karena C[4] adalah 5, maka B[5] diisi dengan 4, dan C[4] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | - | 3 | 4 | 4 | 4 | - |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 2 | 3 | 4 | 7 | 8 |

1 2 3 4 5 6

Langkah 7 : elemen A[2] adalah 6, maka karena C[6] adalah 8, maka B[8] diisi dengan 6, dan C[6] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | - | 3 | 4 | 4 | 4 | 6 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 2 | 3 | 4 | 7 | 7 |

1 2 3 4 5 6

Langkah 8 : elemen A[1] adalah 3, maka karena C[3] adalah 3, maka B[3] diisi dengan 3, dan C[3] dikurangi 1.

A

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 4 | 1 | 3 | 4 | 1 | 4 |

1 2 3 4 5 6 7 8

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 3 | 3 | 4 | 4 | 4 | 6 |

1 2 3 4 5 6 7 8

C

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 2 | 2 | 4 | 7 | 7 |

1 2 3 4 5 6

**Analisis Kompleksitas**

Counting\_Sort(A,B,k)

1. for i to k do k kali
2. C[i] <- 0 k kali
3. for j <- 1 to n do n kali
4. C[A[j]] <- C[A[j]]+1; n kali
5. for i <- 2 to k do k kali
6. C[i] <- C[i] + C[i-1] k kali
7. for j <- n downto 1 do n kali
8. B[C[A[j]]] <- A[j] n kali
9. C[A[j]] <- C[A[j]]-1 n kali

Waktu yang dibutuhkan untuk mengurutkan data menggunakan *counting sort* :

* Loop pertama membutuhkan waktu O(k)
* Loop kedua membutuhkan waktu O(n)
* Loop ketiga membutuhkan waktu O(k), dan
* Loop keempat membutuhkan waktu O(n).

Sehingga kompleksitasnya :

T(n) = O(k) + O(n) + O(k) + O(n)

= O(k) + O(k) + O(n) + O(n)

= O(max(k,k)) + O(max(n,n))

= O(k) + O(n)

T(n) = O(k+n)

dan bisa disimpulkan kompleksitasnya : O(n)