TUGAS 4 KELAS

RUNNING TIME

Disusun sebagai salah satu tugas

mata kuliah Analisis Algoritma



Tasya Amanda Adinegara – 140810160003

Shofiyyah Nadhiroh – 14018160057

Patricia Joanne – 140810160065

Dikumpulkan tanggal

8 April 2019

PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2019

**Soal**

1. Algoritma dan program Heap Sort
2. Gambarkan penjelasan algoritma dengan array 6 elemen.

**Jawaban**

Algoritma Heap Sort:

Heap sort adalah teknik penyortiran berbasis perbandingan yang didasarkan pada struktur data Binary Heap yang mirip dengan pemilihan jenis dimana kita pertama kali menemukan elemen maksimum dan menempatkan elemen maksimum di akhir lalu ulangi proses yang sama untuk elemen yang tersisa.

Binary Heap adalah Binary Tree dimana item disimpan dalam urutan khusus sedemikian rupa sehingga nilai dalam simpul parent lebih besar (atau lebih kecil) dari nilai-nilai dalam dua node anak-anaknya. Item pertama disebut sebagai heap max dan yang terakhir disebut min heap. Heap dapat diwakili oleh pohon biner atau array.

Penerapan dalam array 6 elemen:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Elemen ke- | 0 | 1 | 2 | 3 | 4 | 5 |
| Elemen | 34 | 25 | 22 | 56 | 89 | 12 |

* Iterasi 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 34 | 25 | 22 | 56 | 89 | 12 |
| 34 | 89 | 22 | 56 | 25 | 12 |

* Iterasi 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 34 | 89 | 22 | 56 | 25 | 12 |
| 89 | 34 | 22 | 56 | 25 | 12 |

* Iterasi 3 🡪 Max heap

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 89 | 34 | 22 | 56 | 25 | 12 |
| 89 | 56 | 22 | 34 | 25 | 12 |

* Iterasi 4 🡪 tukar elemen awal dan akhir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 89 | 56 | 22 | 34 | 25 | 12 |
| 12 | 56 | 22 | 34 | 25 | 89 |

* Iterasi 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 12 | 56 | 22 | 34 | 25 | 89 |
| 22 | 56 | 12 | 34 | 25 | 89 |

* Iterasi 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 22 | 56 | 12 | 34 | 25 | 89 |
| 56 | 22 | 12 | 34 | 25 | 89 |

* Iterasi 7 🡪 Max heap

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 56 | 22 | 12 | 34 | 25 | 89 |
| 56 | 34 | 12 | 25 | 22 | 89 |

* Iterasi 8 🡪 tukar elemen awal dan akhir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 56 | 34 | 12 | 25 | 22 | 89 |
| 22 | 34 | 12 | 25 | 56 | 89 |

* Iterasi 9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 22 | 34 | 12 | 25 | 56 | 89 |
| 34 | 22 | 12 | 25 | 56 | 89 |

* Iterasi 10 🡪 Max heap

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 34 | 22 | 12 | 25 | 56 | 89 |
| 34 | 25 | 12 | 22 | 56 | 89 |

* Iterasi 11 🡪 tukar elemen awal dan akhir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 34 | 25 | 12 | 22 | 56 | 89 |
| 22 | 25 | 12 | 34 | 56 | 89 |

* Iterasi 12 🡪 Max heap

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 22 | 25 | 12 | 34 | 56 | 89 |
| 25 | 22 | 12 | 34 | 56 | 89 |

* Iterasi 13 🡪 tukar elemen awal dan akhir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 25 | 22 | 12 | 34 | 56 | 89 |
| 12 | 22 | 25 | 34 | 56 | 89 |

* Iterasi 14 🡪 Max heap

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 12 | 22 | 25 | 34 | 56 | 89 |
| 22 | 12 | 25 | 34 | 56 | 89 |

* Iterasi 15 🡪 tukar elemen awal dan akhir

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 22 | 12 | 25 | 34 | 56 | 89 |
| 12 | 22 | 25 | 34 | 56 | 89 |

Program: [cpp.sh/8xffh](http://cpp.sh/8xffh)

#include <iostream>

#include <chrono>

using namespace std;

using namespace std::chrono;

void heapify(int arr[], int n, int i){

int largest = i;

int l = 2\*i + 1;

int r = 2\*i + 2;

if (l < n && arr[l] > arr[largest])

largest = l;

if (r < n && arr[r] > arr[largest])

largest = r;

if (largest != i){

swap(arr[i], arr[largest]);

heapify(arr, n, largest);

}

}

void heapSort(int arr[], int n){

for (int i = n / 2 - 1; i >= 0; i--)

heapify(arr, n, i);

for (int i=n-1; i>=0; i--){

swap(arr[0], arr[i]);

heapify(arr, i, 0);

}

}

void printArray(int arr[], int n){

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

cout << arr[i] << " ";

}

int main(){

high\_resolution\_clock::time\_point t1 =

high\_resolution\_clock ::now();

int arr[] = {34, 25, 22, 56, 89, 12};

int n = sizeof(arr)/sizeof(arr[0]);

cout << "Array before heap sort: ";

printArray(arr, n);

cout<<endl;

heapSort(arr, n);

cout << "Array after heap sort: ";

printArray(arr, n);

cout<<endl<<endl;

high\_resolution\_clock::time\_point t2 =

high\_resolution\_clock::now();

auto duration = duration\_cast<microseconds>( t2 - t1 ).count();

cout<<duration<<" microseconds"<<endl;

}