**PRAKTIKUM PERANCANGAN & ANALISIS ALGORITMA**

LABORATORIUM INFORMATIKA

UNIVERSITAS GUNADARMA

|  |  |  |
| --- | --- | --- |
| NAMA  NPM  KELAS | : : : | Calvin John Harsono  50422348  3IA04 |
| **a. Ujian terdiri dari dua bagian:**  • **SoalWajib :** Soal-soal yang diberikanwajib dikerjakantanpaterkecuali. Total nilai 50 poin  • **Soal Pilihan:** Soal-soal yang diberikan dapat dikerjakan sesuai pilihan. Boleh dikerjakan semua, boleh tidak jika soal yang dikerjakan sudahmemilikipoin yang cukup.  Total nilai poin soal adalah 100 poin. Maksimum poin yang didapat adalah 50 poin.  **b. Petunjuk pengerjaan:**  o Tenangkan diri dan membaca doa sebelummengerjakan.  o Jawaban langsung diketik di bawah setiap soal.  o Untuk soal isikoderumpang, langsung diisikan pada kode yang diberikan.  o Tidak diperkenankanuntukbekerja sama, mencarijawaban di *Seach Engine* atau menggunakan bantuan AI. Jika melanggar ada sanksi yang akan  diberikan.  o Window pada komputerhanya diperkenankanuntukmembuka website praktikum, aplikasi Dev C++ / compiler online, dan Microsoft Word.  o Format penamaan pengumpulan: **Ujian\_Nama Lengkap\_NPM.pdf**  o **Selamat mengerjakan!** | | |

**SoalWajib (Total 50 Poin)**

1. jelaskan apa itu algoritma greedy ! (10 poin)

- algoritma greedy adalah algoritmayangrakus, tamak dan semacamnya, sifatnya adalah mengambil yang terbesar/paling menguntungkan pada saat itutanpamemikirkan

kedepannya.

2.jelaskann carakerja algoritma divide and concueror ! (15 poin)

- algoritma divide and conqueror berkerja dengan cara membagi permasalahan menjadi

bentuk paling kecilnya(divide), baru setelahnya diselesaikan permasalahan tersebut(conquer), dan Ketikamasing masing permasalahanterkecilnya sudah di selesaikan baru semuanya

digabungkan Kembali

3. apa itu algoritma bruteforce dan contohpemakaiannya? (15 poin)

- algoritma bruteforce adalah algoritmayang sifatnya lurus/lempeng/straight forward,

contohnya adalahpada pencocokan string, algoritma akan membandingkantiap character

pada substring(kata yang ingin di cek) ke string utama(kalimat ataukata yang ingin dicari apakah ada substringnya) sampai semua character pada substring ditemukan pada string utama atautidak ditemukan sama sekali

4. jelaskan perbedaan carakerja algoritma dfs (depth first search) dengan bfs (breath fisrt search) ! (10 poin)

Algoritma DFS melakukantranversal kebagianbawah dahulu,jadi dia akan menelusuri

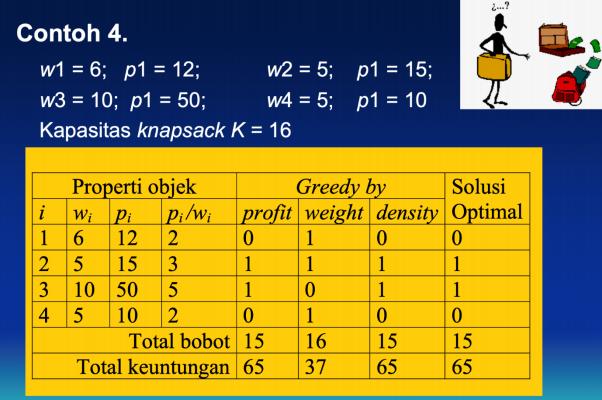
cabang pohon sampaike anak terakhirpadabagian kiri dahulubarumengecek kesampingnya.

Sedangkan BFS akan melakukantranversal menyelesaikan 1 level dahulubaru dia akan mulai menelusuri level selanjutnya. Jadi jika DFS sifat penelusurannya kebawah, BFS penelusurannya kesamping/mendatar

Soal Pilihan (Total 100 poin)

1. Lengkapicodesupaya outputnyamemiliki hasil yang sesuaidengan digambar (25 poin)

// isi yang berwarna merah



#include <iostream>

#include <vector>

#include <algorithm>

using namespacestd;

// Strukturuntuk menyimpan item

struct Item {

int weight;

int value;

double density;

int index;

};

// Fungsi menghitung densitas

void calculateDensity(vector<Item>& items) {

for (auto& item : items) {

item.density = static\_cast<double>(item.value) / item.weight;

}

}

// Fungsi Greedy untuk menyelesaikan knapsack

double knapsackGreedy(vector<Item> items, int maxWeight, int& totalWeightUsed,

vector<Item>& selectedItems, string criteria) {

calculateDensity(items);

// Sorting

if (criteria == <”weight”>) {

sort(items.begin(), items.end(), [](Item a, Item b) {

return a.weight < b.weight;

});

} elseif (criteria == "profit") {

sort(<items.begin()>, items.end(), [](Item a, Item b) {

return a.value > b.value;

});

} elseif (<criteria> == "density") {

< sort >(items.begin(), items.end(), [](Item a, Item b) {

return a.density > b.density;

});

}

double totalValue = 0.0;

totalWeightUsed = 0;

for (const auto& item : items) {

if (totalWeightUsed + item.weight <= maxWeight) {

totalWeightUsed += item.weight;

totalValue += item.value;

selectedItems.push\_back(item);

}

}

return totalValue;

}

// Menampilkan hasil

void displayKnapsackResults(vector<Item>& items, int maxWeight, string criteria) { < cout > << "Hasil Knapsack berdasarkan " << criteria << ":\n";

double totalProfit = 0.0;

inttotalWeightUsed = 0;

vector<Item> selectedItems;

totalProfit = knapsackGreedy(items, maxWeight, totalWeightUsed, selectedItems, criteria);

cout << "Keuntungan maksimum yang dapat diperoleh: " << totalProfit << endl; cout << "Total bobot yang digunakan: " << totalWeightUsed << endl;

cout << "Urutan item yang dipilih: ";

for (const auto& item : selectedItems) {

cout << "(Item " << item.index << " - Berat: " << item.weight << ", Keuntungan: " << item.value << ") ";

}

cout << endl;

cout << " \n";

}

//data yang diketahui

vector<Item> items = {

{6, 12, 0, 1}, // item 1 (weight = 100, value = 40, index = 01)

{5, 15, 0, 2}, // item 2 (weight = 50, value = 35, index = 02)

{10, 50, 0, 3},

{5, 10, 0, 4},

};

int main() {

// int maxWeight = 100; //total kapasitas

vector<Item> items = {

{100, 40, 0, 1}, // item 1 (weight = 100, value = 40, index = 01)

{50, 35, 0, 2}, // item 2 (weight = 50, value = 35, index = 02)

{45, 18, 0, 3},

{20, 4, 0, 4},

{10, 10, 0, 5},

{5, 2, 0, 6}

};

displayKnapsackResults(items, maxWeight, "weight");

displayKnapsackResults(items, maxWeight, "profit");

displayKnapsackResults(items, maxWeight, "density");

return 0;

}

2. Jelaskan code berikut ! (20 poin)

#include<stdio.h>

inta[100];

intmax, min;

void maxmin();

int main() {

inti, num;

printf("\n\t\t\tMaximum Dan Minimum\n\n");

printf("Masukkan Banyak Angka: ");

scanf("%d", &num);

printf("\nMasukkan Angka-angkanya: \n");

for (i = 0; i < num; i++) {

scanf("%d", &a[i]);

}

printf("\nAngka-angkanya adalah: \n");

for (i = 0; i < num; i++) {

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

}

max = a[0];

min = a[0];

maxmin(0,num - 1);

printf("\n\nMaksimum Angka: %d\n", max);

printf("Minimum Angka: %d\n", min);

return 0;

}

void maxmin(inti, intj) {

int max1, min1, mid;

if (i == j) {

max = min = a[i];

} elseif (i == j - 1) {

if (a[i] > a[j]) {

max = a[i];

min = a[j];

} else {

max = a[j];

min = a[i];

}

} else {

mid = (i + j) / 2;

maxmin(i, mid);

max1 = max;

min1 = min;

maxmin(mid + 1, j);

if (max < max1)

max = max1;

if (min > min1)

min = min1;

}

}

Jawab :

-Kodedibawahadalah kode untuk algoritma divide and conquer, dimanauserakandiminta memasukkanangkaangkayangakandicariangka minimum dan maksimumdarisemua

angka yang dimasukkanoleh user. Carakerja kodenya adalah dengan cara membagi 2

rangkaianangka tersebut sampaitersisa 2 atau pun 1 angkauntuk dibandingkan mana yang lebih besar, setelahitubaru mulaidigabungkan Kembali sampaiditemukan mana yang

paling kecildanyang paling besardarisemuaangka tersebut.

3. Jelaskan code berikut ! (15 poin)

#include <iostream>

#include <string>

using namespacestd;

// Fungsi untuk pencocokan string menggunakanalgoritma Brute Force

void bruteForceStringMatch(const string &text, const string &pattern) {

intn = text.length();

intm = pattern.length();

bool found = false;

// Geser pattern di sepanjang text

for (inti = 0; i <= n - m; i++) {

intj;

// Cocokkan karakter satu per satu

for (j = 0; j < m; j++) {

if (text[i + j] != pattern[j]) {

break; // Jika karaktertidak cocok, keluardari loop

}

}

// Jika semua karakter cocok

if (j == m) {

cout << "Pattern ditemukandi indeks " << i << endl;

found = true;

}

}

if (!found) {

cout << "Pattern tidak ditemukanditeks." << endl;

}

}

int main() {

string text, pattern;

cout << "Masukkanteks: ";

getline(cin, text);

cout << "Masukkan pattern yang ingindicari: ";

getline(cin, pattern);

// Panggil fungsi Brute Force

bruteForceStringMatch(text, pattern);

return 0;

}

Jawab:

- kode tersebut adalah kodealgoritma bruteforce untuk melakukan pencocokan string,

dimana user akan memasukan string dan pola yang ingindicek di string. Pada fungsi

bruteForceStringMatch program akan melakukan pengecekkan pola ke stringsecara satu

persatu, dimana jika pola tidak cocok maka pengecekkanakan bergeser ke kanan 1 index

untuk melakukan pengecekkan berikutnya. Dan padaakhirnya program akan melakukan

pengecekan sampaiditemukan kecocokanatau string memangtidak mengandung pola yang ingindicek(tidak ditemukan kesamaan)

4. lengkapi codedan // isi yang berwarnamerah (25 poin) DATA YANG HILANG {while,void,for,queue,true,tetangga)

#include <iostream>

#include <queue>

#include <stack>

#include <vector>

using namespacestd;

class Graf {

private:

int vertices;

vector<vector<int> > adjacencyList;

public:

Graf(intv) : vertices(v), adjacencyList(v + 1) {}

< void > tambahSisi(intv, intw) {

adjacencyList[v].push\_back(w);

adjacencyList[w].push\_back(v); // If the graph is undirected, add this line. }

void BFS(int mulai) {

vector<bool> dikunjungi(vertices + 1, false);

< vector ><int> q;

q.push(mulai);

dikunjungi[mulai] = true;

<while> (!q.empty()) {

intsaatIni = q.front();

cout << saatIni << " ";

q.pop();

// Reverse the order of neighbors

<for> (vector<int>::iterator it = adjacencyList[saatIni].begin(); it !=

adjacencyList[saatIni].end(); ++it) {

inttetangga = \*it;

if (!dikunjungi[tetangga]) {

q.push(tetangga);

dikunjungi[tetangga] = true;

}

}

}

}

void DFS(int mulai) {

vector<bool> dikunjungi(vertices + 1, false);

stack<int> s;

s.push(mulai);

while (!s.empty()) {

intsaatIni = s.top();

s.pop();

if (!dikunjungi[saatIni]) {

cout << saatIni << " ";

dikunjungi[saatIni] = <true>;

}

// Reverse the order of neighbors

for (vector<int>::reverse\_iterator it = adjacencyList[saatIni].rbegin(); it != adjacencyList[saatIni].rend(); ++it) {

inttetangga = \*it;

if (!dikunjungi[tetangga]) {

s.push(<tetangga>);

}

}

}

}

};

int main() {

intvertices, edges;

cout << "Masukkanjumlah simpul: ";

cin >> vertices;

Graf graf(vertices);

cout << "Masukkanjumlahsisi: ";

cin >> edges;

cout << "Masukkan sisi-sisi (format: sumber tujuan):" << endl;

for (inti = 0; i < edges; ++i) {

intsumber, tujuan;

cin >> sumber >> tujuan;

graf.tambahSisi(sumber, tujuan); }

int pilihan;

do {

cout << "\nMenu:\n1. Breadth-First Search (BFS)\n2. Depth-First Search (DFS)\n0. Keluar\nMasukkan pilihan: ";

cin >> pilihan;

switch (pilihan) {

case 1:

cout << "Masukkan simpulawal untuk BFS: ";

intbfsMulai;

cin >> bfsMulai;

cout << "Hasil traversal BFS: ";

graf.BFS(bfsMulai);

cout << endl;

break;

case 2:

cout << "Masukkan simpulawal untuk DFS: ";

intdfsMulai;

cin >> dfsMulai;

cout << "Hasil traversal DFS: ";

graf.DFS(dfsMulai);

cout << endl;

break;

case 0:

cout << "Keluardari program. Selamattinggal!\n";

break;

default:

cout << "Pilihantidak valid. Harapmasukkanopsi yang benar.\n"; } } while (pilihan != 0);

return 0;

}