**Tugas Praktikum Stable Matchi Problem**

Disusun untuk memenuhi tugas praktikum mata kuliah Analisis Logaritma,



Muhammad Rifqy Aulia Akbar

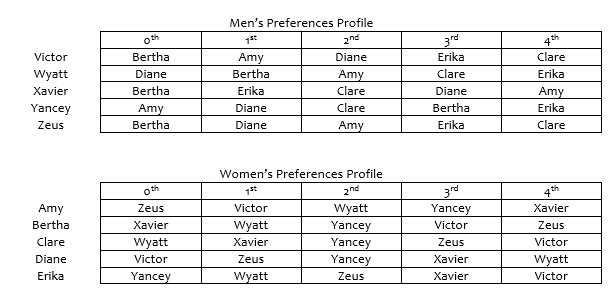
140810160055

Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran

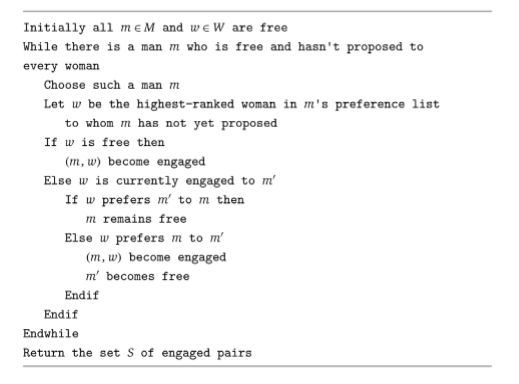
2018

**Soal Worksheet 1**

Dengan Algoritma Gale-Shapley, cari himpunan stable-matching yang sesuai dengan preferencelists berikut ini. Gunakan processor terhebat yang Anda miliki (otak) untuk mengikuti algoritma GS dan output tidak perlu diuraikan per-looping tetapi Anda harus memahami hasil setiap looping.



Algoritma :



Hasil :

Looping 1

* Victor belum bertunangan
* Wyatt bertunangan dengan Diane
* Xavier bertunangan dengan Bertha
* Yancey bertunganan dengan Amy
* Zeus belum bertunangan

Looping 2

* Victor bertunangan dengan Amy
* Wyatt belum bertunangan
* Xavier bertunangan dengan Bertha
* Yancey belum bertunangan
* Zeus bertunangan dengan Diane

Looping 3

* Victor bertunangan dengan Amy
* Wyatt belum bertunangan
* Xavier bertunangan dengan Bertha
* Yancey bertunangan dengan Clare
* Zeus bertunangan dengan Diane

Looping 4

* Victor bertunangan dengan Amy
* Wyatt bertunangan dengan Clare
* Xavier bertunangan dengan Bertha
* Yancey belum bertunangan
* Zeus bertunangan dengan Diane

Looping 5 (selesai)

* Victor menikah dengan Amy
* Wyatt menikah dengan Clare
* Xavier menikah dengan Bertha
* Yancey menikah dengan Erika
* Zeus menikah dengan Diane

**Tugas Praktikum**



#include <iostream>

#include <string.h>

#include <cstring>

#include <string>

using namespace std;

struct orang{

string nama;

bool menikah = false;

int preferensi[5];

string pasangan;

int indexPasangan;

};

string findPerson(int i){

switch(i){

case 0 : return "Amy";

break;

case 1 : return "Bertha";

break;

case 2 : return "Clare";

break;

case 3 : return "Diane";

break;

case 4 : return "Erika";

break;

case 5 : return "Victor";

break;

case 6 : return "Wyatt";

break;

case 7 : return "Xavier";

break;

case 8 : return "Yancey";

break;

case 9 : return "Zeus";

break;

}

}

void deklarasi (orang (&lakiLaki)[5], orang (&perempuan)[5]){

lakiLaki[0].nama = "Victor";

lakiLaki[0].preferensi[0] = 1;

lakiLaki[0].preferensi[1] = 0;

lakiLaki[0].preferensi[2] = 3;

lakiLaki[0].preferensi[3] = 4;

lakiLaki[0].preferensi[4] = 2;

lakiLaki[1].nama = "Wyatt";

lakiLaki[1].preferensi[0] = 3;

lakiLaki[1].preferensi[1] = 1;

lakiLaki[1].preferensi[2] = 0;

lakiLaki[1].preferensi[3] = 2;

lakiLaki[1].preferensi[4] = 4;

lakiLaki[2].nama = "Xavier";

lakiLaki[2].preferensi[0] = 1;

lakiLaki[2].preferensi[1] = 4;

lakiLaki[2].preferensi[2] = 2;

lakiLaki[2].preferensi[3] = 3;

lakiLaki[2].preferensi[4] = 0;

lakiLaki[3].nama = "Yancey";

lakiLaki[3].preferensi[0] = 0;

lakiLaki[3].preferensi[1] = 3;

lakiLaki[3].preferensi[2] = 2;

lakiLaki[3].preferensi[3] = 1;

lakiLaki[3].preferensi[4] = 4;

lakiLaki[4].nama = "Zeus";

lakiLaki[4].preferensi[0] = 1;

lakiLaki[4].preferensi[1] = 3;

lakiLaki[4].preferensi[2] = 0;

lakiLaki[4].preferensi[3] = 4;

lakiLaki[4].preferensi[4] = 2;

perempuan[0].nama = "Amy";

perempuan[0].preferensi[0] = 9;

perempuan[0].preferensi[1] = 5;

perempuan[0].preferensi[2] = 6;

perempuan[0].preferensi[3] = 8;

perempuan[0].preferensi[4] = 7;

perempuan[1].nama = "Bertha";

perempuan[1].preferensi[0] = 7;

perempuan[1].preferensi[1] = 6;

perempuan[1].preferensi[2] = 8;

perempuan[1].preferensi[3] = 5;

perempuan[1].preferensi[4] = 9;

perempuan[2].nama = "Clare";

perempuan[2].preferensi[0] = 6;

perempuan[2].preferensi[1] = 7;

perempuan[2].preferensi[2] = 8;

perempuan[2].preferensi[3] = 9;

perempuan[2].preferensi[4] = 5;

perempuan[3].nama = "Diane";

perempuan[3].preferensi[0] = 5;

perempuan[3].preferensi[1] = 9;

perempuan[3].preferensi[2] = 8;

perempuan[3].preferensi[3] = 7;

perempuan[3].preferensi[4] = 6;

perempuan[4].nama = "Erika";

perempuan[4].preferensi[0] = 8;

perempuan[4].preferensi[1] = 6;

perempuan[4].preferensi[2] = 9;

perempuan[4].preferensi[3] = 7;

perempuan[4].preferensi[4] = 5;

};

int findArray(orang person[], string nama, int indexOrang){

int index;

string indexNama;

int dia;

for(int i = 0; i<5; i++){

index = person[indexOrang].preferensi[i];

indexNama = findPerson(index);

if(indexNama.compare(nama)==0){

dia = i;

}

}

return dia;

};

void hasil(orang lakiLaki[5]){

cout<<"Hasil SMP :"<<endl;

for(int i = 0; i<5; i++){

cout<<lakiLaki[i].nama<<" menikah dengan "<<lakiLaki[i].pasangan<<endl;

}

cout<<endl;

};

void matching(orang (&lakiLaki)[5], orang (&perempuan)[5]){

int temp;

int indexLaki;

string nama1, nama2;

for(int i = 0; i<5; i++){

for(int j = 0; j<5; j++){

if(!lakiLaki[j].menikah){

temp = lakiLaki[j].preferensi[i];

if(!perempuan[lakiLaki[j].preferensi[i]].menikah){

lakiLaki[j].pasangan = perempuan[lakiLaki[j].preferensi[i]].nama;

lakiLaki[j].menikah = true;

lakiLaki[j].indexPasangan = i;

perempuan[temp].pasangan = lakiLaki[j].nama;

perempuan[temp].menikah= true;

perempuan[temp].indexPasangan = findArray(perempuan, lakiLaki[j].nama, temp);

}

else{

if(findArray(perempuan, lakiLaki[j].nama, temp)<perempuan[temp].indexPasangan){

for(int k = 0; k<5; k++){

nama1 = lakiLaki[k].nama;

nama2 = perempuan[temp].pasangan;

if(nama1.compare(nama2)==0)

indexLaki = k;

}

lakiLaki[indexLaki].menikah = false;

lakiLaki[indexLaki].pasangan = "";

lakiLaki[j].pasangan = perempuan[lakiLaki[j].preferensi[i]].nama;

lakiLaki[j].menikah = true;

lakiLaki[j].indexPasangan = i;

perempuan[temp].pasangan = lakiLaki[j].nama;

perempuan[temp].indexPasangan = findArray(perempuan, lakiLaki[j].nama, temp);

}

}

}

}

}

}

int main()

{

orang cowo[5];

orang cewe[5];

deklarasi(cowo, cewe);

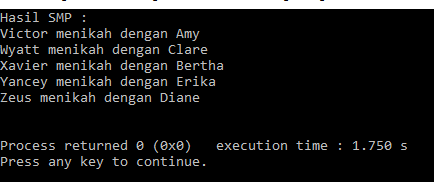
matching(cowo, cewe);

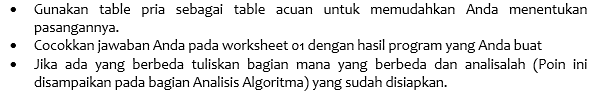
hasil(cowo);

return 0;

}

Output Program :





Tidak ada perbedaan.

**Analisis Algoritma**

* 1. Apakah jawaban Anda di Worksheet 01 dan Program sama persis? Jika Tidak? Kenapa?

Ya jawaban dari worksheet 01 dan program sama persis.

* 1. Anda diminta untuk membuktikan algoritma G-S benar dengan menjawab pertanyaan berikut:
* Fakta (1.1): Seorang wanita tetap bertunangan dari titik di mana dia menerima proposal pertamanya; dan urutan mitra yang bertunangan dengannya menjadi lebih baik dan lebih baik lagi (hal ini sesuai dengan daftar preferensi wanita).

🡪 Tidak perlu dipertanyakan

* Fakta (1.2): Urutan wanita yang dilamar pria lebih buruk dan lebih buruk lagi (hal ini sesuai dengan daftar preferensi pria).

🡪 Tidak perlu dipertanyakan

* Teorema (1.3): Algoritma G-S berakhir setelah paling banyak n2 iterasi menggunakan While Loop. Buktikan!

🡪 dalam kasus ini, n adalah 5, satu laki – laki akan menikahi dengan satu perempuan, dengan kata lain, iterasi minimalnya adalah jumlah pasangan laki – laki dan perempuan, yaitu 5, adapun tingkat kecocokan dimana dapat membuat laki – laki dan perempuan berganti pasangan, hal ini membuat paling banyak satu laki – laki melamar 5 perempuan, jika ada 5 laki – laki, maka iterasi paling banyak adalah 5 (kali pelamaran) x 5 (jumlah laki – laki) yaitu 25 (n2).

* Teorema (1.4): Jika seorang pria bebas di beberapa titik dalam eksekusi algoritma, maka ada seorang wanita yang belum dia ajak bertunangan. Buktikan!

🡪 Dengan syarat Perfect Match, maka 1 laki – laki akan menikahi 1 perempuan, jika jumlah laki – laki ada 5 orang, dan perempuan 5 orang, saat ada laki – laki belum menikah, dipastikan ada perempuan yang belum menikah.

* Teorema (1.5): Himpunan S yang dikembalikan saat terminasi adalah perfect matching. Buktikan!

🡪 hasil dari terminasi adalah

S (V-A, W-C, X-B, Y-E, Z-D)

Dimana setiap laki – laki tepat menikahi 1 perempuan, tidak ada satu pun yang sendiri, ataupun memiliki 2 pasangan.

* Teorema (1.6): Sebuah eksekusi algoritma G-S mengembalikan satu set pasangan S. Set S adalah pasangan yang stabil. Buktikan!

🡪 Terkait dengan fakta 1.1, dan fakta 1.2, maka perempuan akan mendapatkan laki – laki terbaik yang bisa didapatkan, dan laki – laki akan melamar perempuan terbaik sesuai dengan kesukaan dari perempuan, hal ini menghasilkan pasangan yang stabil.