LAPORAN SEDERHANA

SISTEM INFORMASI AKADEMIK



Nama : Setya Handika

NRP : 5024211029

Departemen : Teknik Komputer

Mata Kuliah : Pemrogaman Lanjutan

Kelas : A

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2022

**Pembahasan**

<https://github.com/HandikaSetya70/Tugas-SIM-Akademik-2>

Pada tugas ini, mahasiswa diminta untuk membuat SIM Akademik mulai dari 0. Tugas dikerjakan oleh Setya Handika NRP 5024211029.

Pada hasil ini, dibuatlah beberapa kelas. Yang pertama adalah kelas user\_data, kelas sebagai berikut:

Text

Description automatically generated

Pada kelas ini, didapat member yaitu:

* name : Nama
* dob : Tempat tanggal lahir
* logid : Login id
* pass : Password
* gender : Jenis kelamin

Terdapat pula fungsi dalam kelas, yaitu changepass. Tujuan utama changepass disini adalah sebagai fungsi untuk membantu user mengganti password.

Kemudian, terdapat kelas Mahasiswa. Data dari mahasiswa ini juga mengambil data dari kelas sebelumnya, yaitu user\_data.

class mahasiswa : public user\_data {

    public:

        string NRP;

        string major;

        mahasiswa(){};

Terdapat 2 member untuk mahasiswa, yaitu NRP dan major. NRP adalah NRP dari mahasiswa tersebut, dan major adalah jurusan.

Kemudian terdapat pendeklarasian untuk input data kedalam kelas mahasiwa tersebut:

mahasiswa(string a, string c, string d, string e, string f, string g, string h){

            NRP = a;

            major = c;

            name = d;

            dob = e;

            logid = f;

            pass = g;

            gender = h;

        }

Maka akan mempermudah input untuk kelas mahasiswa. Kemudian terdapat beberapa fungsi di kelas tersebut:

void print(){

            cout << "Data Mahasiswa" << endl << endl;

            cout << "Nama          : " << name << endl;

            cout << "Date of birth : " << dob << endl;

            cout << "Gender        : " << gender << endl;

            cout << "NRP           : " << NRP << endl;

            cout << "Major         : " << major << endl;

            cout << "logid         : " << logid << endl;

            cout << "pass          : " << pass << endl;

            return;

        }

Fungsi diatas, print, bertujuan untuk mengoutputkan data mahasiswa.

void input(){

            string name\_a;

            string dob\_a;

            string logid\_a;

            string pass\_a;

            string gender\_a;

            string NRP\_a;

            string major\_a;

            cin.ignore();

            cout << "Name           : ";

            getline(cin, name\_a);

            cout << "Date of birth  : ";

            getline(cin, dob\_a);

            cout << "Create Login ID: ";

            cin >> logid\_a;

            cout << "Create password: ";

            cin >> pass\_a;

            cout << "Gender         : ";

            cin >> gender\_a;

            cout << "NRP            : ";

            cin >> NRP\_a;

            cin.ignore();

            cout << "Major          : ";

            getline(cin, major\_a);

            name = name\_a;

            dob = dob\_a;

            logid = logid\_a;

            pass = pass\_a;

            gender = gender\_a;

            NRP = NRP\_a;

            major = major\_a;

            return;

        }

Fungsi diatas, input, bertujuan untuk memasukan data namun secara manual dan bertahap.

Kemudian terdapat kelas tendik atau tenaga kependidikan.

class tendik : public user\_data {

    public:

        string profession;

        string ID;

        int year\_of\_entry;

        string education;

        int jml\_kelas;

        tendik(){};

        tendik(string a, int b, string c, string d, string e, string f, string g, string h, string i){

            ID = a;

            year\_of\_entry = b;

            education = c;

            name = d;

            dob = e;

            logid = f;

            pass = g;

            gender = h;

            profession = i;

        }

        void print(){

            cout << "Data Tenaga Kependidikan" << endl << endl;

            cout << "Nama          : " << name << endl;

            cout << "Date of birth : " << dob << endl;

            cout << "Gender        : " << gender << endl;

            cout << "ID            : " << ID << endl;

            cout << "Year of entry : " << year\_of\_entry << endl;

            cout << "Education     : " << education << endl;

            cout << "logid         : " << logid << endl;

            cout << "pass          : " << pass << endl;

            return;

        }

        void input(){

            string name\_a;

            string dob\_a;

            string logid\_a;

            string pass\_a;

            string gender\_a;

            string ID\_a;

            double year\_of\_entry\_a;

            string education\_a;

            string profession\_a;

            cin.ignore();

            cout << "Name           : ";

            getline(cin, name\_a);

            cout << "Date of birth  : ";

            getline(cin, dob\_a);

            cout << "Create Login ID: ";

            cin >> logid\_a;

            cout << "Create password: ";

            cin >> pass\_a;

            cout << "Gender         : ";

            cin >> gender\_a;

            cout << "ID             : ";

            cin >> ID\_a;

            cout << "Year of entry  : ";

            cin >> year\_of\_entry\_a;

            cin.ignore();

            cout << "Education      : ";

            getline(cin, education\_a);

            cout << "Profession     : ";

            cin >> profession\_a;

            name = name\_a;

            dob = dob\_a;

            logid = logid\_a;

            pass = pass\_a;

            gender = gender\_a;

            ID = ID\_a;

            year\_of\_entry = year\_of\_entry\_a;

            education = education\_a;

            profession = profession\_a;

            return;

        }

};

Tenaga kependidikan memiliki tambahan member:

* profession : profesi
* ID : ID
* year\_of\_entry : tahun masuk
* education : Pendidikan
* jml\_kelas : Jumlah kelas dari tendik

Kelas tersebut juga memiliki fungsi yang tidak jauh berbeda dengan kelas mahasiswa.

Kemudian terdapat kelas croom (classroom).

class croom{

    public:

        int jumlah\_mhs = 0;

        tendik dosen;

        string subjek;

        map<string,mahasiswa> Mhs;

        map<string,int> nilai;

        //string: NRP

        croom(){}

        croom(tendik a, string b){

            dosen = a;

            subjek = b;

        }

Pada kelas ini terdapat beberapa member:

* jumlah\_mhs : Jumlah mahasiswa
* dosen : Data dosen (dengan kelas tendik)
* subjek : Subjek dari kelas tersebut
* Mhs : List mahasiswa dengan indeks merupakan NRP nilai : List nilai dari mahasiswa dengan indeks merupakan NRP

Kemudian terdapat beberapa fungsi,

        int getgradebynrp(string k){

            if(Mhs[k].name==""){

                return 0;

            }

            return nilai[k];

        }

        void showgradebynrp(string k){

            if(Mhs[k].name==""){

                return;

            }

            cout << endl;

            cout << "Class " << subjek << endl;

            cout << "Name  : " << Mhs[k].name << endl;

            cout << "NRP   : " << Mhs[k].NRP << endl;

            cout << "Grade : " << nilai[k] << endl;

            return;

        }

Kedua fungsi ini memiliki fungsi yang sama, yaitu untuk mendapatkan nilai, namun fungsi yang pertama guna mendapat nilai return berupa nilai, kemudian fungsi kedua menampilkan data secara full, yaitu nama, nrp, dan nilai.

        void input\_mhs(mahasiswa a){

            Mhs[a.NRP] = a;

            nilai[a.NRP] = 0;

            jumlah\_mhs++;

            return;

        }

Fungsi input\_mhs ini berguna untuk menambahkan mahasiswa dalam kelas tersebut.

        void get\_mhs(){

            map<string,mahasiswa>::iterator it;

            map<string,int>::iterator ti;

            cout << "Jumlah mahasiswa   : " << jumlah\_mhs << endl;

            int k = 1;

            for(it = Mhs.begin(), ti = nilai.begin(); it != Mhs.end() && k <= jumlah\_mhs; ti++, it++, k++){

                cout << k << ". " << it->second.NRP << "   " << ti->second << "  \t" << it->second.name << endl;

            }

            cout << endl;

            return;

        }

Fungsi diatas berguna untuk menampilkan jumlah mahasiswa dan data dari mahasiswa kelas tersebut.

        void input\_nilaiauto(string k, int bro){ //string = NRP

            int be = 0;

            nilai[k] = bro;

            return;

        }

        void input\_nilamanual(){

            int n;

            string b;

            get\_mhs();

            cout << "NRP nilai yang di input: ";

            cin >> b;

            cout << "Jumlah nilai: ";

            cin >> n;

            ;

            nilai[b] = n;

            cout << "Input successful!\nName: " << Mhs[b].name << "\nGrade: " << nilai[b] << endl;

            return;

        }

Kedua fungsi tersebut berguna untuk menginput nilai. Fungsi pertama secara otomatis dan kedua secara manual.

        void getinfo(){

            cout << "Kelas              : " << subjek << endl;

            cout << "Dosen              : " << dosen.name << endl;

            return;

        }

Fungsi diatas getInfo adalah untuk menampilkan informasi tentang kelas yaitu subjek dan dosen.

Kemudian untuk main menu awal, maka ditampilkan 3 menu

Text

Description automatically generated with low confidence

Pilihan 1 untuk **login**, pilihan 2 untuk **membuat user baru**, dan pilihan 3 untuk **exit**.

Untuk pilihan 1, maka ditampilkan sebagai berikut:

Text

Description automatically generated

Akan diminta input berupa ID dan password.

Apabila salah, maka muncul

Text

Description automatically generated

Dengan kode:

    cout << "ID       : ";

    cin >> gol;

    cout << "Password : ";

    cin >> pas;

    if(pas == dataMhs[gol].pass){

        cout << "login successful!\n";

        bru = true; //mhs

    } else if(pas == dataTendik[gol].pass){

        cout << "login successful!\n";

        bru = false;//tendik

    } else if(k<3){

        cout << "incorrect ID / password. Please try again!\n";

        k++;

        pause();

        goto blok;

    } else {

        cout << "You failed to login 3x. Please try again later.\n";

        pause();

        exit(1);

    }

Memanfaatkan fungsi if dan else if, maka akan di cek password tersebut sesuai atau tidak.

Apabila selesai login, maka muncul tampilan sebagai berikut:

Text

Description automatically generated

Menu utama memanfaatkan switch case dengan case sesuai dengan fungsinya masing-masing.

Untuk **create new user**, memanfaatkan fungsi input dalam kelas mahasiswa dan tendik. Apabila dipilih menu pertama, akan muncul

Text

Description automatically generated

Maka akan dapat membuat *mahasiswa* maupun *tendik*.

Kemudian untuk pilihan **user information**, akan menampilkan informasi akun user tersebut. Contoh sebagai berikut:

Text

Description automatically generated

Kemudian untuk pilihan ketiga yaitu **change user password**, tampilannya sebagai berikut:

Text

Description automatically generated

Akan diminta input password baru berupa string. Kemudian setelah berhasil diganti akan ditampilkan seperti diatas. Kodenya:

        case 3:

        {

            if (bru){

                dataMhs[gol].changepass();

            } else {

                dataTendik[gol].changepass();

            }

            break;

        }

Memanfaatkan fungsi changepass pada kelas *user\_data*.

Kemudian terdapat pilihan **classroom**. Pada pilihan ini, terdapat 2 macam layout. Apabila anda adalah *mahasiswa*, maka tampilan sebagai berikut:

A screenshot of a computer

Description automatically generated with medium confidence

Pada pilihan pertama, **show classes**, berfungsi untuk menunjukan semua kelas yang ada. Contoh tampilan adalah sebagai berikut:

Text

Description automatically generated

Kode dari pilihan diatas adalah

                vector<croom>::iterator it;

                int b = 0;

                it = datacroom.begin();

                while(it!=datacroom.end()){

                    cout << "Class number " << b+1 << endl;

                    it->getinfo();

                    it->get\_mhs();

                    b++;

                    it++;

                }

Memanfaatkan iterator terhadap data classroom yang telah di declare sebelumnya,

    vector<croom> datacroom;

maka akan menunjukan semua classroom yang ada.

Kemudian pilihan kedua, **join a class** (requires permission), dengan tampilan sebagai berikut:

Text

Description automatically generated

Akan ditampilkan semua kelas kemudian diminta input berupa nomor kelas dimana user ingin gabung. Kodenya adalah sebagai berikut

                vector<croom>::iterator it;

                int bro = 0, in, b = 0;

                it = datacroom.begin();

                while(it!=datacroom.end()){

                    map<string,mahasiswa>::iterator w;

                    for(w = it->Mhs.begin(); w != it->Mhs.end(); w++){

                        if(dataMhs[gol].NRP == w->first){

                            goto kek;

                        }

                    }

                    cout << "Class number " << b+1 << endl;

                    it->getinfo();

                    it->get\_mhs();

                    bro++;

                    kek:

                    b++;

                    it++;

                }

                if(bro == 0){

                    cout << "\nYou are already a student of all available class!\n" << endl;

                } else {

                    cout << "Which class do you want to join? Please input the class number: ";

                    cin >> in;

                    datpermit[gol] = datacroom[in-1];

                    clear();

                    logo();

                    cout << "\nRequest to join class sent!\nPlease wait until confirmation!\n Class info:\n Class number " << in << endl;

                    datacroom[in-1].getinfo();

                    datacroom[in-1].get\_mhs();

                    cout << endl;

                }

Maka dalam kode tersebut, digunakan beberapa perulangan seperti while dan for. Kemudian memanfaatkan pointer yang dideklarasi dengan cara berikut.

                map<string,mahasiswa>::iterator w;

Inti dari kode diatas adalah untuk mengecek apakah mahasiswa tersebut sudah berada di dalam kelas tersebut atau belum, apabila belum maka akan dapat dilanjutkan dengan menampilkan kelas tersebut sebagai salah satu pilhan. Apabila tidak terdapat kelas yang ditampilkan maka didapat tampilan:

A screenshot of a computer screen

Description automatically generated with medium confidence

Kemudian untuk pilihan **show my grades**, akan ditampilkan sebagai berikut:

Text

Description automatically generated

Dengan kode:

                int ind = 0;

                double ip = 0;

                int jmlip = 0;

                vector<croom>::iterator klok = datacroom.begin();

                while(klok!=datacroom.end()){

                    klok->showgradebynrp(dataMhs[gol].NRP);

                    int b = datacroom[ind].getgradebynrp(dataMhs[gol].NRP);

                    if(b!=0){

                        ip += grade(b);

                        jmlip++;

                    }

                    ind++;

                    klok++;

                }

                cout <<"\n IP: " << ip/jmlip << endl << endl;

Maka pada kode tersebut, menggunakan perulangan untuk mencari nilai dari semua kelas terkhusus nilai dari user tersebut, menggunakan fungsi showgradebynrp dan getgradebynrp. Kemudian akan dihitung sehingga didapat indeks IP, dan menggunakan fungsi untuk menampilkan IP menggunakan kode:

double grade(int b){

    double k;

    if(b >= 86){

        k = 4;

    } else if(b> 75 && b < 86){

        k = 3.5;

    } else if(b > 65 && b < 76){

        k = 3.2;

    } else if(b > 60 && b < 66){

        k = 3;

    } else if(b > 55 & b < 61){

        k = 2.7;

    } else if(b > 40 && b < 56){

        k = 2.4;

    } else if(b < 41){

        k = 2;

    }

    return k;

}

Kemudian untuk tampilan **classroom** sebagai *tendik* sebagai berikut:

Text

Description automatically generated

Pilihan pertama sama dengan mahasiswa, yaitu menampilkan kelas. Pilihan kedua, **Create a new class**, untuk membuat kelas baru. Tampilan sebagai berikut:

Text

Description automatically generated

Dengan kode:

                string subj;

                cout << "Please input class's subject: ";

                cin >> subj;

                croom blob(dataTendik[gol],subj);

                datacroom.push\_back(blob);

                cout << "Success!\n Class info:\n";

                blob.getinfo();

                blob.get\_mhs();

Maka memanfaatkan fungsi push\_back, menambahkan data *croom* pada datacroom. Kemudian setelah selesai dibuat kelas baru, maka data kelas baru tersebut akan ditampilkan.

Kemudian untuk pilihan ketiga, **input grades**, akan didapat tampilan:

Text

Description automatically generated

Dengan kode:

                int x, b = 0;

                clear();

                logo();

                vector<croom>::iterator it;

                for(it = datacroom.begin(); it != datacroom.end(); it++, b++){

                    if(it->dosen.ID != dataTendik[gol].ID){

                        continue;

                    }

                    cout << "\nClass number " << b+1 << endl;

                    it->getinfo();

                    it->get\_mhs();

                }

                cout << "Which class number? \n Option: ";

                cin >> x;

                datacroom[x-1].input\_nilamanual();

Maka memanfaatkan fungsi dari kelas *croom* yaitu *input\_nilaimanual* untuk menginput nilai secara manual.

Setelah input nilai selesai maka akan ditampilkan:

Text

Description automatically generated

Kemudian pilihan 4, **Class admission request**, apabila terdapat request dengan kelas yang bersangkutan dengan user (tendik) akan ditampilkan:

Text

Description automatically generated

Apabila tidak maka akan ditampilkan:

A screenshot of a computer

Description automatically generated with low confidence

Dengan kode:

                clear();

                map<string,croom>::iterator it;

                int ok, ko = 0;

                for(it = datpermit.begin(); it != datpermit.end(); it++){

                    if(it->second.dosen.ID==dataTendik[gol].ID){

                        cout << "Request number    : " << ko+1 << endl;

                        cout << " Student name     : " << dataMhs[it->first].name << endl;

                        cout << " Student name     : " << dataMhs[it->first].NRP << endl;

                        cout << " Class subject    : " << it->second.subjek << endl;

                        cout << endl;

                        ko++;

                    }

                }

                if(ko==0){

                    clear();

                    logo();

                    cout << endl;

                    cout << "There are no request made." << endl;

                    pause();

                    goto exfile;

                }

                cout << " Request number to confirm : ";

                cin >> ok;

                it = datpermit.begin();

                for(int j = 0; j != ok-1 && it != datpermit.end() ; it++){

                    if(it->second.dosen.ID==dataTendik[gol].ID){

                        j++;

                    }

                }

                clear();

                logo();

                vector<croom>::iterator lo;

                for(lo = datacroom.begin(); lo->subjek != it->second.subjek; lo++){};

                lo->input\_mhs(dataMhs[it->first]);

                datpermit.erase(it->first);

                cout << "Student permitted!" << endl;

                cout << "Class info: " << endl;

                lo->getinfo();

                lo->get\_mhs();

Apabila terdapat request dengan dosen terkait, maka akan ditampilkan data yang bersangkutan, kemudian terdapat pilihan dengan tampilan:

Text

Description automatically generated

Maka mahasiswa yang bersangkutan akan berhasil ditambahkan dalam classroom dengan fungsi input mahasiswa pada kelas *croom*.

Kemudian untuk pilihan ke 5 yaitu **back**. Perintah dari pilihan ini adalah simple dengan menggunakan perintah sebagai berikut:

Graphical user interface

Description automatically generated with medium confidence

Kemudian untuk pilihan ke 5 pada menu utama, **logout**, adalah untuk me-logout user dan mengembalikannya ke halaman utama.

Text

Description automatically generated

Dengan kode:

            int ram;

            cout << "Do you want to log out of your account?\n input 0 to go back and 1 to log out.\n input: ";

            cin >> ram;

            if(ram==1){

                cout << "logout successful!\n";

                pause();

                goto blok;

            } else if(ram==0){

                goto exfile;

            }

            pause();

            break;

Apabila sukses logout, maka akan kembali ke menu awal.

Kemudian untuk pilihan ke 6, **exit**, akan mengeluarkan user dari program dengan kode:

            return;