LAPORAN PRAKTIKUM

POSTTEST (1)

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Muhammad Nabil Rahmatullah (2409106046)

Kelas (B1 ‘24)

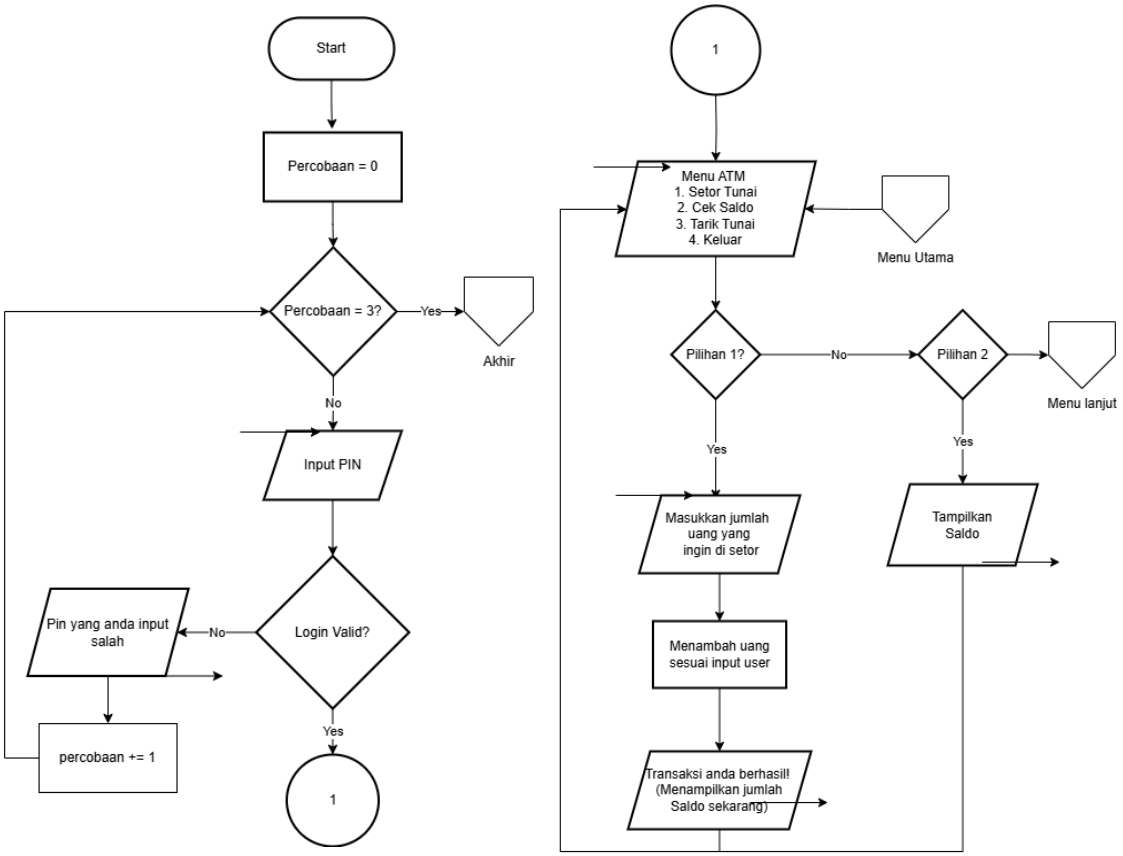
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

SAMARINDA

2025

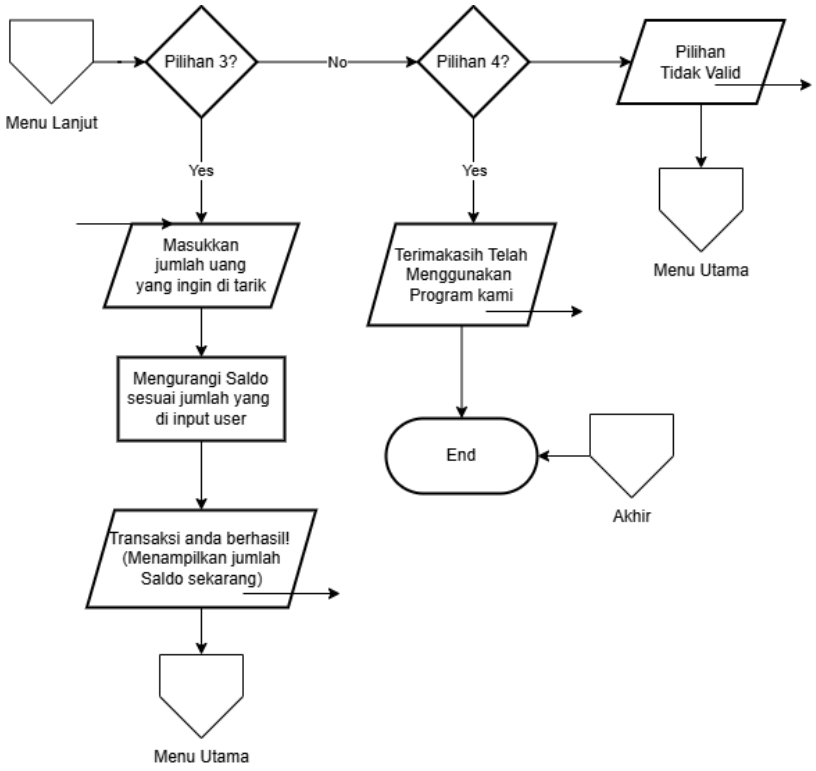
1. Flowchart



Gambar 1.1 Flowchart Login dan menu

Pada Langkah pertama program akan melakukan pengecekkan apakah percobaan sama dengan 3 atau tidak. Jika tidak sama maka program akan meminta *user* untuk melakukan *input* pin untuk melakukan validasi. Jika *login* tersebut dinyatakan valid, maka program akan melanjutkan ke menu utama. Namun jika *login* tersebut tidak valid maka program akan memunculkan *output* "Pin yang anda input salah" dan akan menambah percobaan sebanyak 1 kali, dan program akan kembali ke atas untuk melakukan pengecekkan apakah program sama dengan 3? Jika iya maka program akan otomatis terhenti.

Kemudian jika *user* telah berhasil melakukan *login* maka program akan melanjutkan ke menu utama. Pada menu utama, *user* akan diminta untuk memilih menu mana yang ia inginkan. Jika *user* memilih menu pertama, maka *user* akan diminta melakukan *input* jumlah uang yang ingin di setor ke ATM. Jika *user* memilih menu yang ke dua, maka program akan menampilkan jumlah saldo terkini dari *user* tersebut.



Gambar 1.2 Flowchart Menu

Jika *user* memilih menu ke tiga, maka *user* akan diminta menulis jumlah uang yang ingin di tarik, dan program akan otomatis mengurangi jumlah saldo sesuai dengan uang yang ditarik oleh *user*, kemudian program akan kembali ke Menu Utama. Lalu jika *user* memilih pilihan ke empat maka program akan berhenti. Namun jika *user* melakukan *input* selain angka "1-4" maka program akan memunculkan pesan "Pilihan tidak valid" dan program akan kembali ke menu utama.

2. Analisis Program

2.1 Deskripsi Singkat Program

Tujuan dari program ini adalah untuk meningkatkan efisiensi operasional bank dengan mengurangi ketergantungan pada tenaga kerja manual, sehingga dapat menekan biaya operasional, khususnya dalam pembayaran gaji pegawai. Dengan sistem ini, nasabah dapat melakukan transaksi seperti tarik tunai, setor tunai, dan kedepannya dapat transfer, secara mandiri tanpa perlu mengunjungi teller, sehingga layanan perbankan menjadi lebih cepat dan efisien

2.2 Penjelasan Alur & Algoritma

Pada awal program *user* diberi kesempatan sebanyak tiga kali dengan cara membuat variabel “percobaan = 0” yang akan bertambah sebanyak satu setiap kali *user* gagal *login.* Lalu program akan meminta *user* memasukkan PIN, autentikasi dilakukan dengan membandingkan PIN yang dimasukkan dengan PIN yang benar. Jika *user* gagal login sebanyak tiga kali, maka akun akan di blokir. Setelah *login* berhasil, program menampilkan menu transaksi dalam *loop do-while*. Program menggunakan struktur *if-else* untuk menangani berbagai jenis transaksi.

Program memiliki beberapa mekanisme validasi input:

- Memastikan input menu adalah angka
- Memastikan input jumlah setor/tarik adalah angka
- Memeriksa bahwa jumlah setor/tarik lebih dari 0
- Memeriksa bahwa jumlah tarik tidak melebihi saldo

Program akan berakhir ketika *user* memilih keluar atau akun terblokir karena terlalu banyak percobaan *login* yang gagal.

Fitur Pada Program

1. **Maskering PIN**: Menampilkan bintang (*) saat memasukkan PIN
2. **Batasan Percobaan**: Membatasi percobaan login yang gagal
3. **Validasi Input**: Mencegah input yang tidak valid
4. **Pembersihan Layar**: Membersihkan layar untuk menjaga tampilan tetap rapi

3. Source Code

A. Fitur Clear Terminal

Fitur ini digunakan untuk membersihkan terminal agar pada saat program berjalan akan terlihat lebih rapi. Saya menggunakan *if-else* untuk mengantisipasi error jika *user* menjalankan program pada *windows* maupun *linux*

Source code:

```
void clearTerminal(){
    #ifdef _WIN32
        system("cls");
    #else
        system("clear");
    #endif
}
```

Gambar 3.1 Code Login

B. Fitur Sistem Keamanan Login

Fitur ini berfungsi untuk menghindari aksi pembobolan oleh oknum yang tidak bertanggung jawab. Pada fitur ini saya memberi kesempatan untuk *login* sebanyak tiga kali. Jika *user* mencoba *login* lebih dari tiga kali, maka akun akan terblokir secara otomatis.

Dan juga saya menggunakan “getch()” agar pada saat *user* melakukan *input* PIN di terminal hanya terlihat simbol “*” yang bertujuan untuk mencegah aksi pembobolan.

Source code:

```
int main() {
    string PIN = "6046";

    float saldo = 100;

    int percobaan = 0;
    const int maxPercobaan = 3;

    string inputPIN;
    char ch;
    bool loginSuc = false;
```

```
    while (percobaan < maxPercobaan){
        inputPIN = "";
        cout << "Masukkan PIN : ";

        while (true) {
            ch = getch();
            if (ch == 13){
                cout << endl;
                break;
            } else if (ch == 8 && inputPIN.length() > 0) {
                inputPIN.pop_back();
                cout << "\b \b";

            } else if (ch >= '0' && ch <= '9' && inputPIN.length() < 4){
                inputPIN.push_back(ch);
                cout << "*";
            }
        }

        if (inputPIN == PIN){
            cout << "Login Berhasil!" << endl;
            loginSuc = true;
            break;

        } else {
            percobaan++;
            cout << "Pin Salah! Sisa percobaan " << percobaan << "/" <<
maxPercobaan << endl;
        }
    }
    if (!loginSuc){
        cout<<"Akun Anda terblokir, silahkan hubungin call center!" << endl;
        return 0;
    }
```

Gambar 3.2 Fitur Keamanan

C. Fitur Validasi Input

Fitur ini bertujuan agar *user* tidak melakukan *input* diluar dari yang kita inginkan, contohnya pada saat *user* ingin tarik tunai maupun setor tunai, program hanya akan menerima *input* angka. Jika *user* memaksa melakukan input selain angka (simbol,huruf, dll) maka program akan memberi pesan "Input tidak valid! Silahkan masukkan angka". Disini saya menggunakan "**system('pause')**" agar sebelum terminal dibersihkan, *user* diminta menekan tombol apapun yang ada di *keyboard* mereka

Source code:

```
void clearInput(){
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
}


if (cin.fail() || cin.peek() != '\n'){
    cout << "Input tidak Valid! Silahkan input angka." << endl;
    clearInput();
    system("pause");
    continue;
}
```

Gambar 3.3 Fitur Validasi

D. Fitur Transaksi

Fitur ini berfungsi untuk melakukan berbagai transaksi keuangan seperti, tarik tunai dan setor tunai. Disini saya menggunakan “cin.fail” dan “cin.peek” untuk melakukan validasi input. Dan saya menggunakan percabangan *if-else* untuk menangani berbagai kasus seperti, *user* melakukan *input* selain angka dan *user* melakukan *input* kurang dari 1. Jika *user* melakukan *input* dengan benar, maka program menambahkan saldo sesuai dengan jumlah yang disetor oleh *user*. Dan program akan memberitahu bahwa transaksi berhasil atau tidak.

Source code:

```
    float setor;
    cout << "Masukkan Jumlah yang ingin anda setor (IDR): ";
    cin >> setor;
    if (cin.fail() || cin.peek() != '\n'){
        cout << "Input tidak Valid! Silahkan input angka." << endl;
        clearInput();
        system("pause");
        continue;
    }

    else if (setor > 0) {
        saldo += setor;
        cout << endl;
        tampilkanGaris(50);
        cout << "            TRANSAKSI BERHASIL            " << endl;
        cout << "    Saldo Anda Sekarang: Rp " << saldo << endl;
        tampilkanGaris(50);
```

```
    } else {
        cout << endl;
        tampilkanGaris(50);
        cout << "  Jumlah yang disetor harus lebih dari Rp 0  " << endl;
        tampilkanGaris(50);
    }
}
```

Gambar 3.4 Fitur Setor Tunai

4. Uji Coba dan Hasil Output

4.1 Uji Coba

Untuk memastikan program berjalan dengan baik dan sesuai dengan keinginan kita, dilakukan beberapa skenario pengujian dengan berbagai jenis input. Berikut adalah beberapa skenario pengujian yang diterapkan:

Skenario 1: Pengujian Login dengan PIN

- **Input:**
    - Memasukkan PIN yang benar.
    - Memasukkan PIN yang salah hingga batas maksimum percobaan.
- **Ekspektasi Hasil:**
    - Jika PIN benar, pengguna dapat mengakses menu utama.
    - Jika PIN salah hingga 3 kali, akun akan terblokir dan program berhenti.

Skenario 2: Pengujian Menu Pilihan Transaksi

- **Input:**
    - Memilih opsi menu dengan angka di luar rentang 1-4.
    - Memasukkan input non-numerik.
- **Ekspektasi Hasil:**
    - Jika memilih opsi yang valid, program akan menjalankan fitur terkait.
    - Jika input tidak valid, program menampilkan pesan kesalahan dan meminta input ulang.

Skenario 3: Pengujian Setor Tunai

- **Input:**
    - Memasukkan jumlah setoran yang valid.
    - Memasukkan jumlah setoran negatif atau nol.
    - Memasukkan karakter non-numerik.
- **Ekspektasi Hasil:**
    - Jika jumlah setoran valid, saldo bertambah sesuai nominal yang dimasukkan.
    - Jika jumlah setoran tidak valid, program menampilkan pesan kesalahan dan meminta input ulang.

Skenario 4: Pengujian Cek Saldo

- **Input:**
    - Memilih opsi cek saldo setelah beberapa transaksi.
- **Ekspektasi Hasil:**
    - Program menampilkan saldo terkini sesuai dengan transaksi yang telah dilakukan.

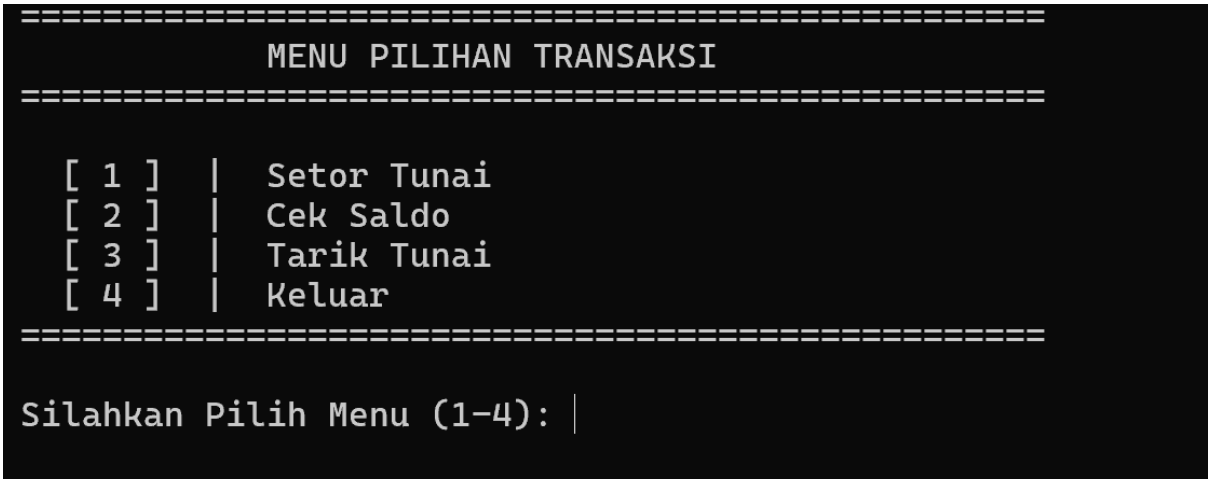
Skenario 5: Pengujian Tarik Tunai

- **Input:**
    - Memasukkan jumlah tarik tunai yang valid (di dalam batas saldo).
    - Memasukkan jumlah yang melebihi saldo.
    - Memasukkan jumlah negatif atau nol.
    - Memasukkan karakter non-numerik.
- **Ekspektasi Hasil:**
    - Jika jumlah tarik tunai valid, saldo berkurang sesuai nominal yang dimasukkan.
    - Jika melebihi saldo, program menampilkan pesan saldo tidak cukup.
    - Jika input tidak valid, program menampilkan pesan kesalahan dan meminta input ulang.

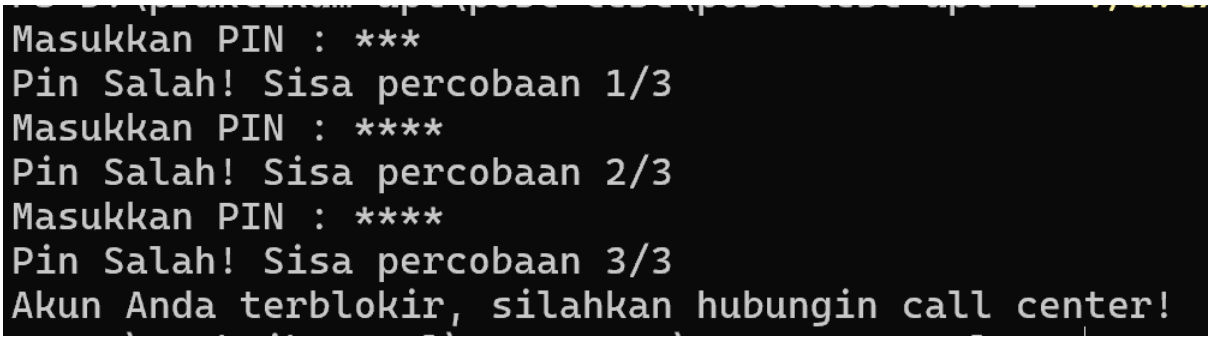
Skenario 6: Pengujian Keluar dari Program

- **Input:**
    - Memilih opsi keluar dari menu utama.
- **Ekspektasi Hasil:**
    - Program menampilkan pesan terima kasih dan berhenti berjalan.

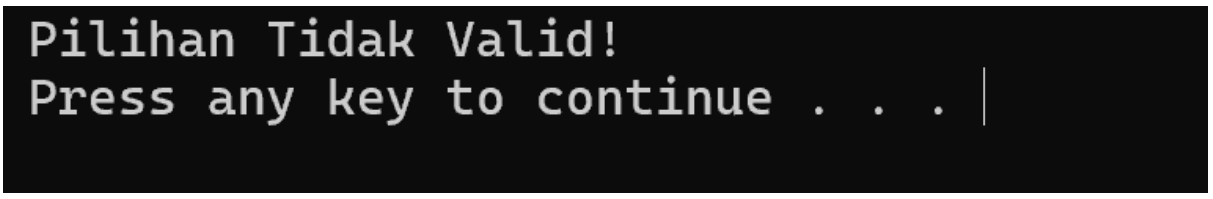
4.2 Hasil Output



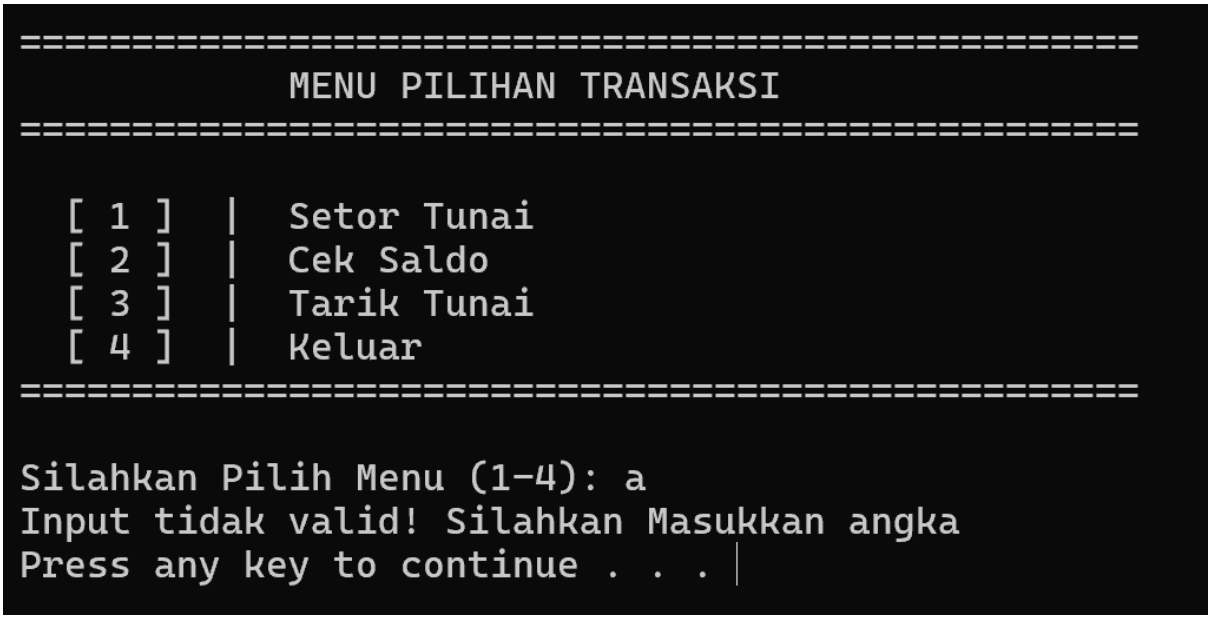
Gambar 4.2.1 Output jika user berhasil login



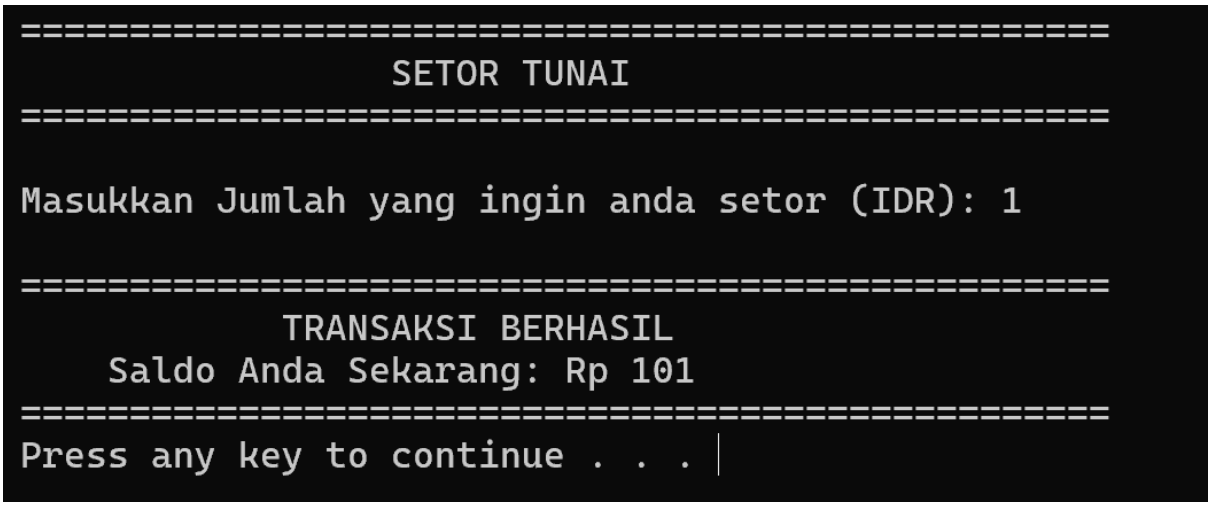
Gambar 4.2.2 Output jika user gagal login sebanyak tiga kali



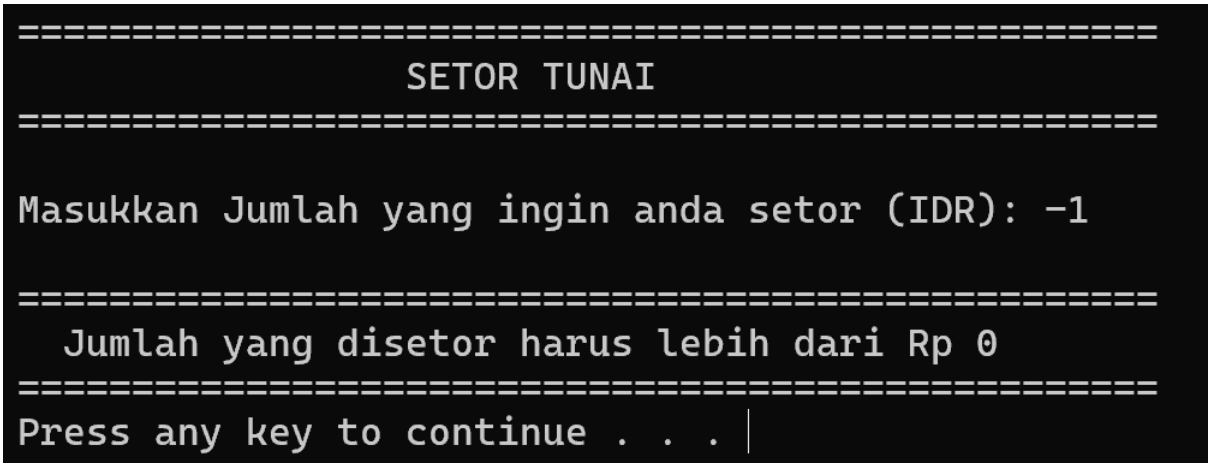
Gambar 4.2.3 Output jika user melakukan input diluar 1-4 pada menu



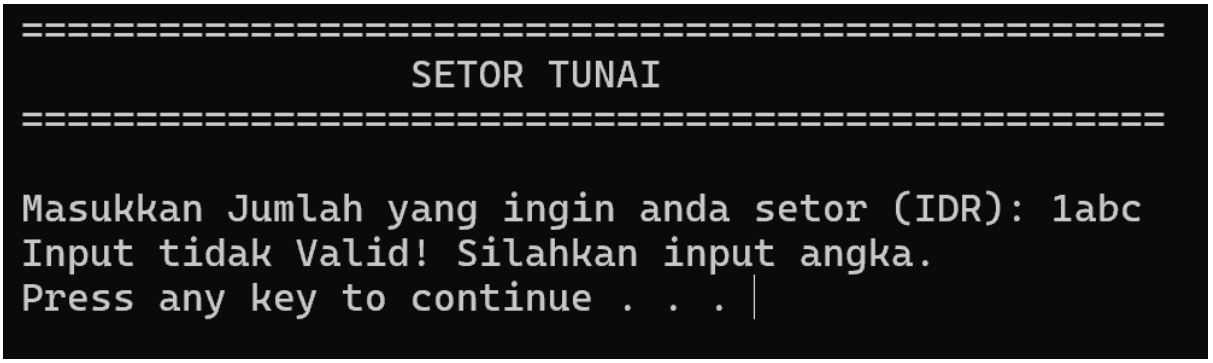
Gambar 4.2.4 Output jika user melakukan input selain angka pada menu



Gambar 4.2.5 Output jika user memasukkan jumlah setoran yang valid

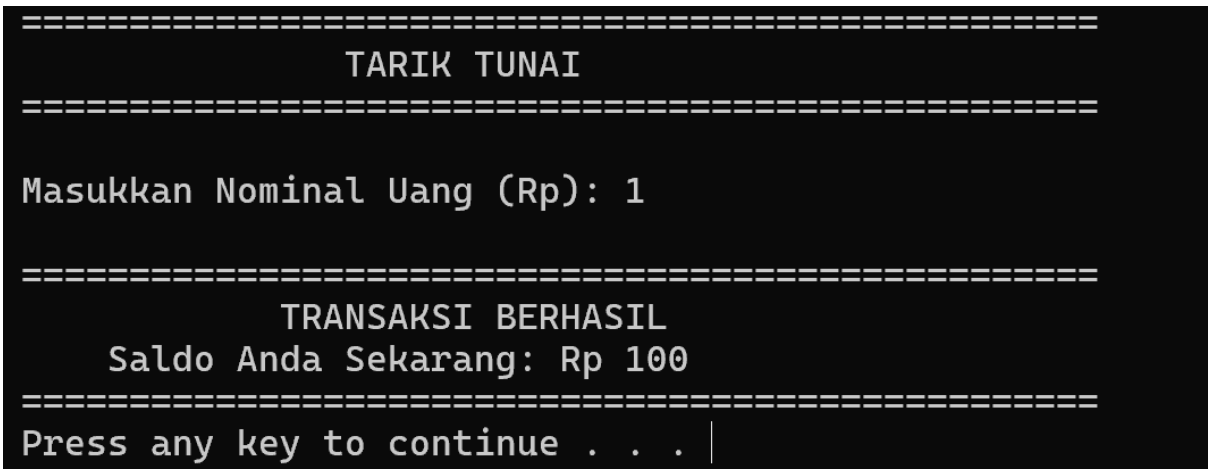


Gambar 4.2.6 Output jika user memasukkan jumlah setoran kurang daris satu rupiah



Gambar 4.2.7 Output jika user memasukkan karakter non-numerik pada saat setor tunai

Gambar 4.2.8 Output jika user melakukan cek saldo



Gambar 4.2.9 Output jika user memasukkan jumlah tarik tunai yang valid

```
==================================================
               TARIK TUNAI
==================================================

Masukkan Nominal Uang (Rp): 101

==================================================
          SALDO ANDA TIDAK CUKUP
==================================================
Press any key to continue . . .
```

Gambar 4.2.10 Output jika user memasukkan jumlah tarik tunai melebihi dari saldo

```
==================================================
               TARIK TUNAI
==================================================

Masukkan Nominal Uang (Rp): -1

==================================================
  Jumlah yang ditarik harus lebih dari Rp 0
==================================================
Press any key to continue . . .
```

Gambar 4.2.11 Output jika user memasukkan jumlah tarik tunai kurang dari 1

```
==================================================
               TARIK TUNAI
==================================================

Masukkan Nominal Uang (Rp): tes
Input tidak valid! Silahkan input angka
Press any key to continue . . .
```

Gambar 4.2.12 Output jika user memasukkan angka non-numerik pada menu tarik tunai

```
==================================================
               TERIMA KASIH
      Telah menggunakan layanan kami
==================================================
```

Gambar 4.2.13 Output jika user memilih opsi keluar dari program

5. Langkah-Langkah GIT

5.1 GIT INIT

Melakukan “git init” untuk menginisiasikan repository git dengan cara menulis “git init” pada directory yang kita inginkan

```
PS D:\praktikum-apl\post-test> cd post-test-apl-1
PS D:\praktikum-apl\post-test\post-test-apl-1> git init
Initialized empty Git repository in D:/praktikum-apl/post-test/post-test-apl-1/.git/
PS D:\praktikum-apl\post-test\post-test-apl-1>
```

Gambar 5.1.1 Melakukan git init

5.2 GIT ADD

Berfungsi untuk menandai *file* mana yang ada perubahan dengan cara menulis “git add “nama file” atau gunakan “.” jika ingin menandai semua file

```
PS D:\praktikum-apl\post-test\post-test-apl-1> git add 2409106046-MuhammadNabilRahmatullah-PT-1.cpp
PS D:\praktikum-apl\post-test\post-test-apl-1> git status
On branch main

No commits yet
```

Gambar 5.2.1 Melakukan git add

5.3 GIT COMMIT

Berfungsi untuk melakukan perubahan pada *file* yang kita pantau dan dapat memberi keterangan apa yang kita ubah pada *file* tersebut. Caranya dengan menulis ‘git commit -m “pesan yang ingin ditulis”'

```
PS D:\praktikum-apl\post-test\post-test-apl-1> git commit -m "Finish Post Test 1"
[main c7a200b] Finish Post Test 1
 1 file changed, 26 insertions(+), 8 deletions(-)
PS D:\praktikum-apl\post-test\post-test-apl-1>
```

Gambar 5.3.1 Melakukan git commit

5.4 GIT REMOTE

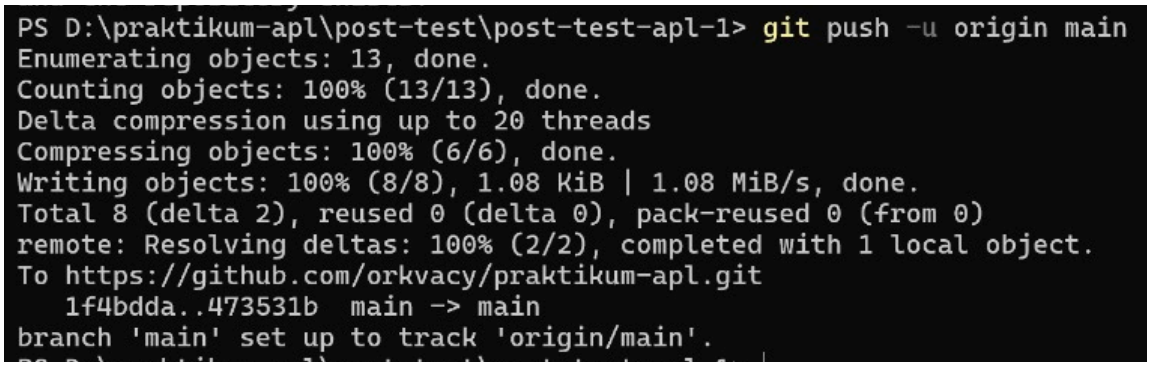
Berfungsi untuk menyambungkan langsung ke repository yang terdapat pada *repository remote (github)* kita.
Dengan cara menulis “git remote add origin link-repo”

```
PS D:\praktikum-apl\post-test\post-test-apl-1> git remote add origin https://github.com/orkvacy/praktikum-apl.git
PS D:\praktikum-apl\post-test\post-test-apl-1> git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 20 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 831 bytes | 831.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/orkvacy/praktikum-apl.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Gambar 5.4.1 Melakukan git remote

5.5 GIT PUSH

Berfungsi untuk melakukan *push* atau mengirimkan rubahan yang telah di *commit* ke *repository remote* (github). Dengan cara menulis “git push -u origin main”



Gambar 5.5.1 melakukan git push