LAPORAN PRAKTIKUM

POSTTEST 1

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Langgeng Dimas

Saputra

(2409106121)

Kelas (C2 ‘24)

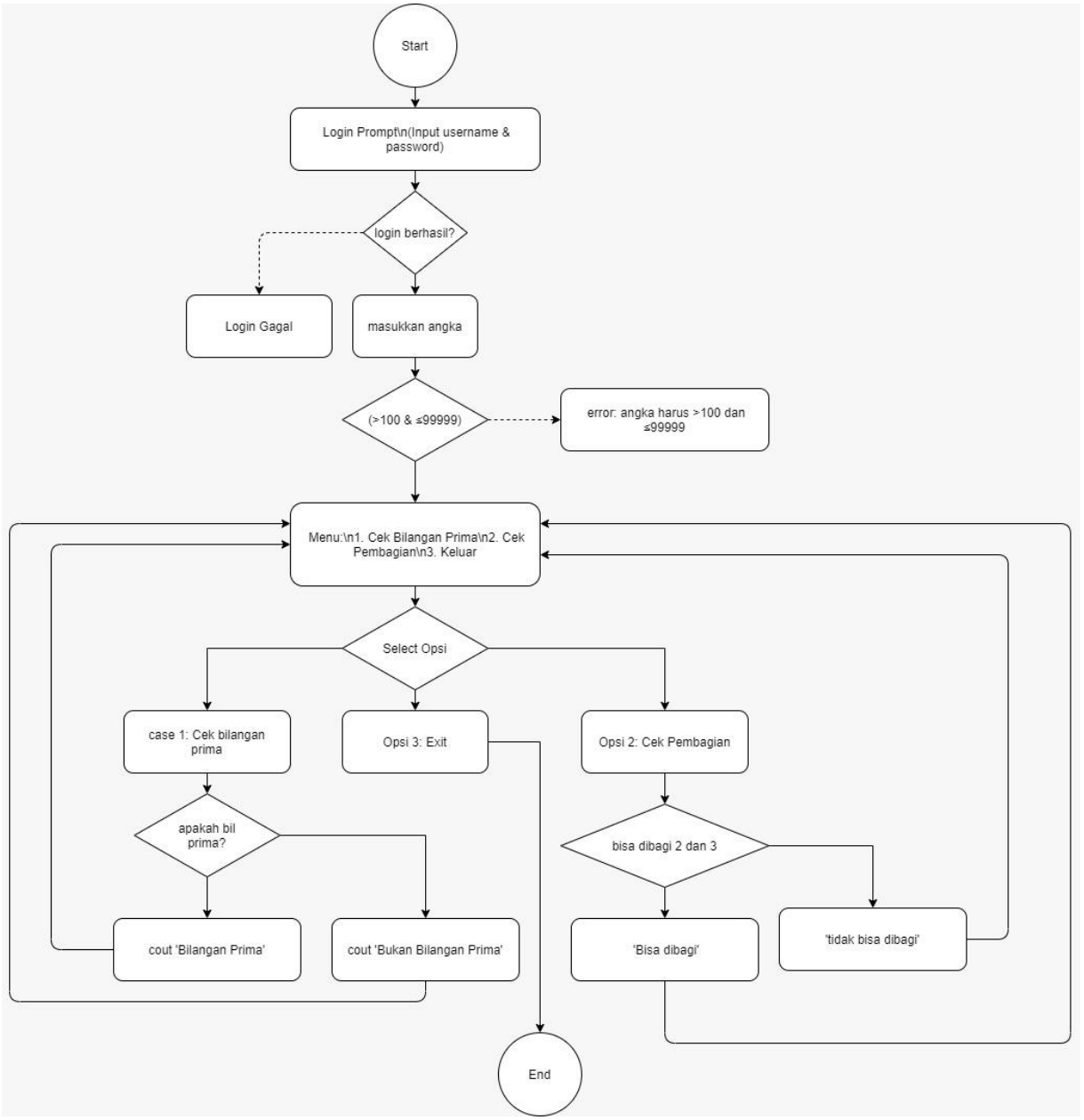
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

SAMARINDA

2025

1. Flowchart


Start
Login Prompt\n(Input username & password)
login berhasil?
Login Gagal
masukkan angka
(>100 & ≤99999)
error: angka harus >100 dan ≤99999
Menu:\n1. Cek Bilangan Prima\n2. Cek Pembagian\n3. Keluar
Select Opsi
case 1: Cek bilangan prima
Opsi 3: Exit
Opsi 2: Cek Pembagian
apakah bil prima?
bisa dibagi 2 dan 3
cout 'Bilangan Prima'
cout 'Bukan Bilangan Prima'
'Bisa dibagi'
'tidak bisa dibagi'
End

2. Analisis Program

2.1 Deskripsi Singkat Program

Tujuan:

Program ini dibuat untuk memberikan sistem login dan validasi kode sandi, serta menyediakan fitur verifikasi angka dengan beberapa metode matematika.

Fungsi/Manfaat Utama:

1. **Sistem Keamanan dengan Login**
   - Program mengharuskan pengguna memasukkan username dan password yang benar sebelum dapat mengakses fitur lainnya.

2. **Validasi Kode Sandi**
   - Memastikan angka yang dimasukkan adalah bilangan bulat positif dalam rentang tertentu (lebih dari 100 dan tidak lebih dari 99999).

3. **Verifikasi Angka dengan Beberapa Metode:**
   - **Cek Bilangan Prima:** Memeriksa apakah angka yang dimasukkan adalah bilangan prima.
   - **Cek Pembagian:** Menentukan apakah angka tersebut habis dibagi 2 dan 3 tetapi tidak habis dibagi 5.

4. **Interaktif dan Berulang**
   - Program menggunakan menu yang memungkinkan pengguna memilih operasi yang ingin dijalankan hingga mereka memilih keluar.

Dengan program ini, pengguna dapat melakukan analisis angka sederhana sambil mendapatkan pengalaman dalam sistem login dan validasi data.

2.2 Penjelasan Alur & Algoritma

A. Penjelasan Alur Program

1. **Proses Login**
    - Program meminta pengguna memasukkan username dan password.
    - Jika sesuai dengan data yang ditentukan, program melanjutkan ke langkah berikutnya.
    - Jika tidak sesuai, program menampilkan pesan "Login gagal!" dan langsung berhenti.
2. **Validasi Input Angka**
    - Setelah login berhasil, pengguna diminta memasukkan angka (bilangan bulat positif).
    - Angka harus berada dalam rentang 101 hingga 99999.
    - Jika angka tidak memenuhi syarat, program menampilkan pesan kesalahan dan meminta pengguna menginput ulang.
3. **Menampilkan Menu Pilihan**
    - Program menyediakan tiga opsi utama, yaitu:
        1. Cek Bilangan Prima
        2. Cek Pembagian (Habis dibagi 2 & 3 tetapi tidak 5)
        3. Keluar
    - Pengguna memilih salah satu opsi, lalu program menjalankan proses verifikasi berdasarkan pilihan tersebut.
4. **Proses Verifikasi**
    - Jika pengguna memilih opsi 1, program akan mengecek apakah angka merupakan bilangan prima dan menampilkan hasilnya.
    - Jika pengguna memilih opsi 2, program akan mengecek apakah angka habis dibagi 2 & 3 tetapi tidak 5, lalu menampilkan hasilnya.
    - Jika pengguna memilih opsi 3, program akan keluar.
5. **Looping Menu**
    - Setelah setiap operasi verifikasi selesai, program kembali menampilkan menu pilihan.
    - Pengguna dapat memilih kembali hingga memilih opsi keluar.

B. Algoritma

1. Mulai program:

2. Proses Login:

- Minta input username dan password dari pengguna.
- Bandingkan dengan username dan password yang telah ditentukan.
- Jika sesuai, lanjut ke proses berikutnya.
- Jika tidak sesuai, tampilkan pesan "Login gagal!" dan hentikan program.

3. Validasi Input Angka:

- Minta pengguna memasukkan bilangan bulat positif.
- Periksa apakah angka berada dalam rentang 101 hingga 99999.
- Jika tidak memenuhi syarat, tampilkan pesan kesalahan dan minta input ulang hingga angka valid.

4. Tampilkan Menu Pilihan:

- Berikan tiga opsi:
    1. Cek Bilangan Prima
    2. Cek Pembagian (Habis dibagi 2 & 3 tetapi tidak 5)
    3. Keluar
- Minta pengguna memilih salah satu opsi.

5. Proses Verifikasi Berdasarkan Pilihan:

- Jika pengguna memilih 1 (Cek Bilangan Prima):
    - Jika angka kurang dari 2, nyatakan bukan bilangan prima.
    - Lakukan pengecekan apakah angka memiliki pembagi selain 1 dan dirinya sendiri:
        - Iterasi dari 2 hingga akar kuadrat dari angka.
        - Jika angka bisa dibagi habis oleh bilangan lain, nyatakan bukan bilangan prima.
        - Jika tidak ada pembagi, nyatakan bilangan prima.
    - Tampilkan hasilnya.
- **Jika pengguna memilih 2 (Cek Pembagian):**
    - Periksa apakah angka habis dibagi 6 (karena $6 = 2 \times 3$).
    - Periksa apakah angka tidak habis dibagi 5.
    - Jika kedua kondisi terpenuhi, tampilkan pesan bahwa angka memenuhi syarat.
    - Jika tidak, tampilkan pesan bahwa angka tidak memenuhi syarat.
- **Jika pengguna memilih 3 (Keluar):**
    - Tampilkan pesan "Keluar."
    - Hentikan program.

6. Ulangi Proses Menu:

- Jika pengguna memilih opsi 1 atau 2, kembali ke menu utama setelah menampilkan hasil verifikasi.
- Jika pengguna memilih opsi 3, program berhenti.

7. Selesai.

3 Source Code

A.Fungsi Bilangan Prima

Tujuan: Mengecek apakah suatu angka merupakan bilangan prima.

```
bool cekBilanganPrima(int angka) {
    if (angka < 2) return false;
    for (int i = 2; i * i <= angka; i++) {
        if (angka % i == 0) return false;
    }
    return true;
}
```

B.Validasi Code

Tujuan: Memastikan angka yang dimasukkan berada dalam rentang valid, yaitu lebih dari 100 dan tidak lebih dari 99999.

```
14  bool validasiKode(int angka) {
15      return (angka > 100 && angka <= 99999);
16  }
17
```

C.Fungsi Pembagian

Tujuan: Mengecek apakah angka habis dibagi 6 (berarti habis dibagi 2 dan 3) dan tidak habis dibagi 5.

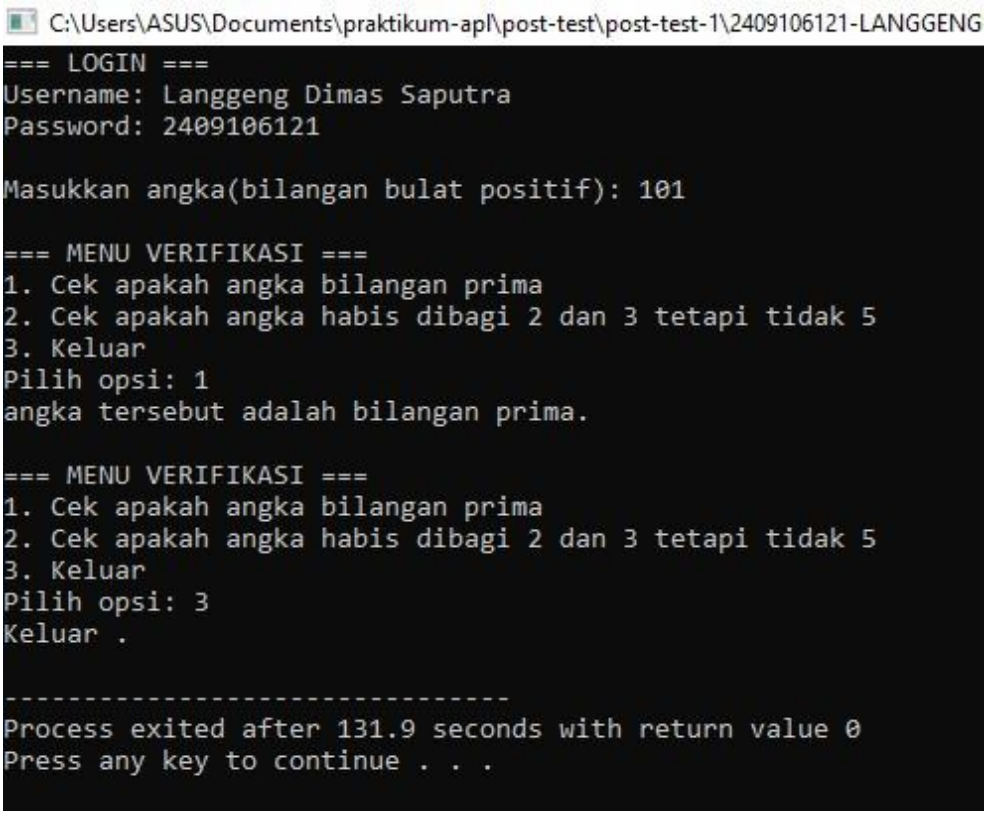
```
bool cekPembagian(int angka) {
    return (angka % 6 == 0 && angka % 5 != 0);
}
```

4.Uji Coba dan Hasil Output

4.1 Uji Coba

- Login:
- Masukkan:
    - Username: Langgeng Dimas Saputra
    - Password: 2409106121
- Hasil: Login berhasil.
- Input Angka:
- Masukkan angka: 101
- Validasi:
    - Karena 101 > 100 dan ≤ 99999, input valid.
- Catatan: 101 merupakan bilangan prima.
- Menu Verifikasi:
- Pilih opsi 1 (cek bilangan prima).
- Hasil:
    - Output: "angka tersebut adalah bilangan prima."

4.2 Hasil Output



5. Git

1. Konfigurasi User Git

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git config --global user.email "langgengdimas121@gmail.com"
```

- Perintah ini digunakan untuk mengatur alamat email yang akan digunakan dalam commit Git.
- Opsi --global berarti pengaturan ini berlaku untuk semua repository di komputer.

2. Inisialisasi Repository Git

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git init
Reinitialized existing Git repository in C:/Users/ASUS/Documents/praktikum-apl/.git/
```

- Perintah ini digunakan untuk menginisialisasi repository Git di dalam folder saat ini.
- Jika repository Git sudah ada, Git akan menampilkan pesan **"Reinitialized existing Git repository"**.

3. Menambahkan Remote Repository

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git remote add origin https://github.com/2409106121langgengdimassaputra/praktikum-apl.git
error: remote origin already exists.
```

- Perintah ini menghubungkan repository lokal dengan repository online di GitHub.
- **origin** adalah nama default untuk remote repository.
- Jika sebelumnya sudah ada remote dengan nama **origin**, maka akan muncul pesan **"remote origin already exists"**.

4. Menambahkan File ke Staging Area

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git add .
```

- Perintah ini digunakan untuk menambahkan semua file yang telah diubah atau ditambahkan ke dalam **staging area** sebelum dilakukan commit.

5. Membuat Commit

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git commit -m "Langgeng Posttest"
On branch master
nothing to commit, working tree clean
```

- Commit digunakan untuk menyimpan perubahan ke dalam repository lokal.
- Opsi -m "Langgeng Posttest" menambahkan pesan deskriptif tentang perubahan yang dilakukan.
- Jika tidak ada perubahan, Git akan menampilkan **"nothing to commit, working tree clean"**.

6. Mengunggah ke Repository GitHub (Push)

```
ASUS@LAPTOP-7JK1V59G MINGW64 ~/Documents/praktikum-apl (master)
$ git push -u origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 446.12 KiB | 808.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/2409106121langgengdimassaputra/praktikum-apl.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

- Perintah ini mengunggah perubahan dari branch **master** di lokal ke branch **master** di remote **origin** (GitHub).
- Opsi -u (--set-upstream) menghubungkan branch lokal dengan branch di remote, sehingga ke depannya cukup menggunakan git push tanpa parameter tambahan.
- Pesan **"Please complete authentication in your browser..."** muncul karena GitHub membutuhkan autentikasi sebelum bisa mengunggah perubahan.
- Setelah autentikasi berhasil, proses *push* berjalan hingga selesai.
- Menunjukkan bahwa file berhasil dikompresi dan dikirim ke GitHub.