LAPORAN PRAKTIKUM

POSTTEST 3

ALGORITMA PEMROGRAMAN LANJUT

Disusun oleh:

Nabila Putri karni (2409106041)

Kelas (A2 ‘24)

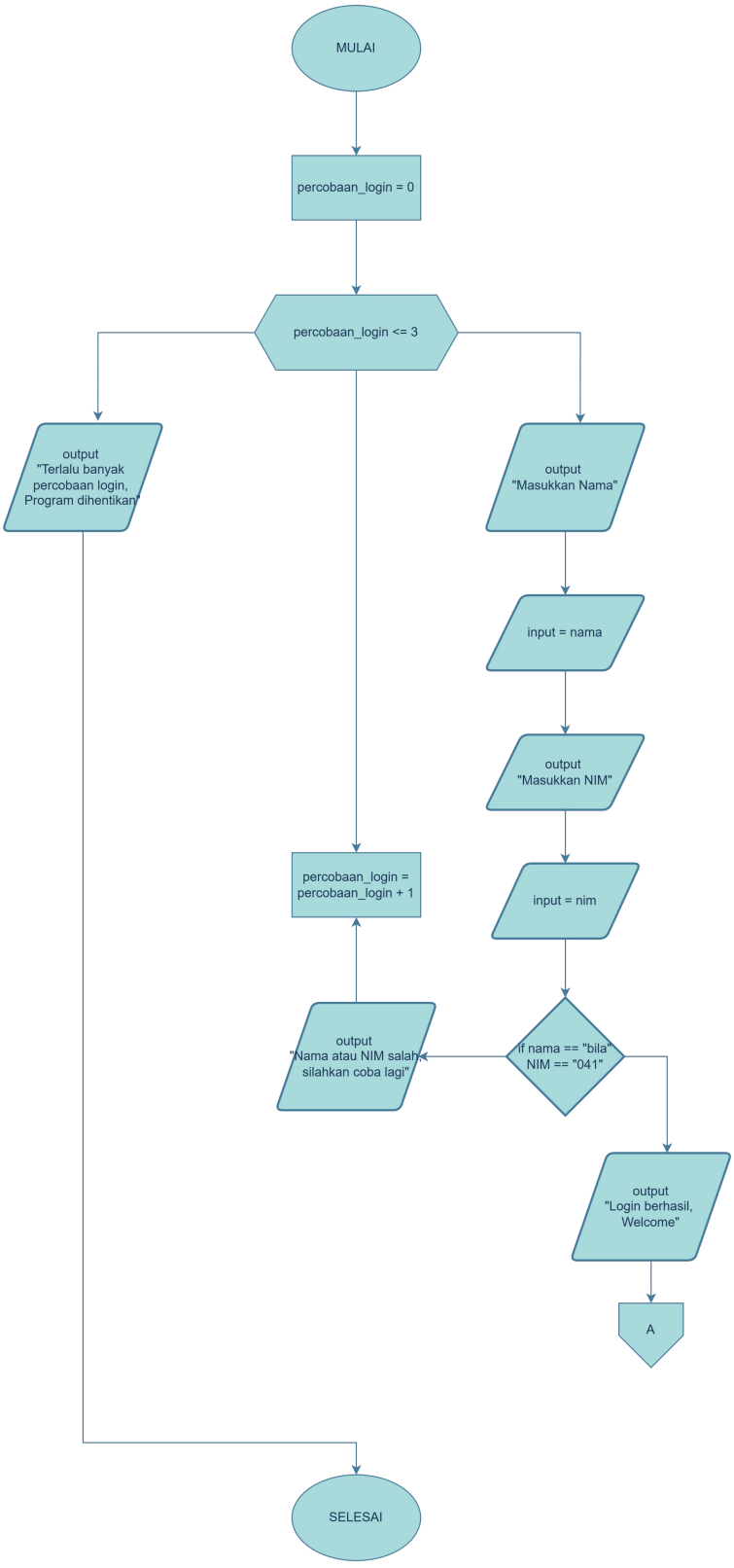
PROGRAM STUDI INFORMATIKA

UNIVERSITAS MULAWARMAN

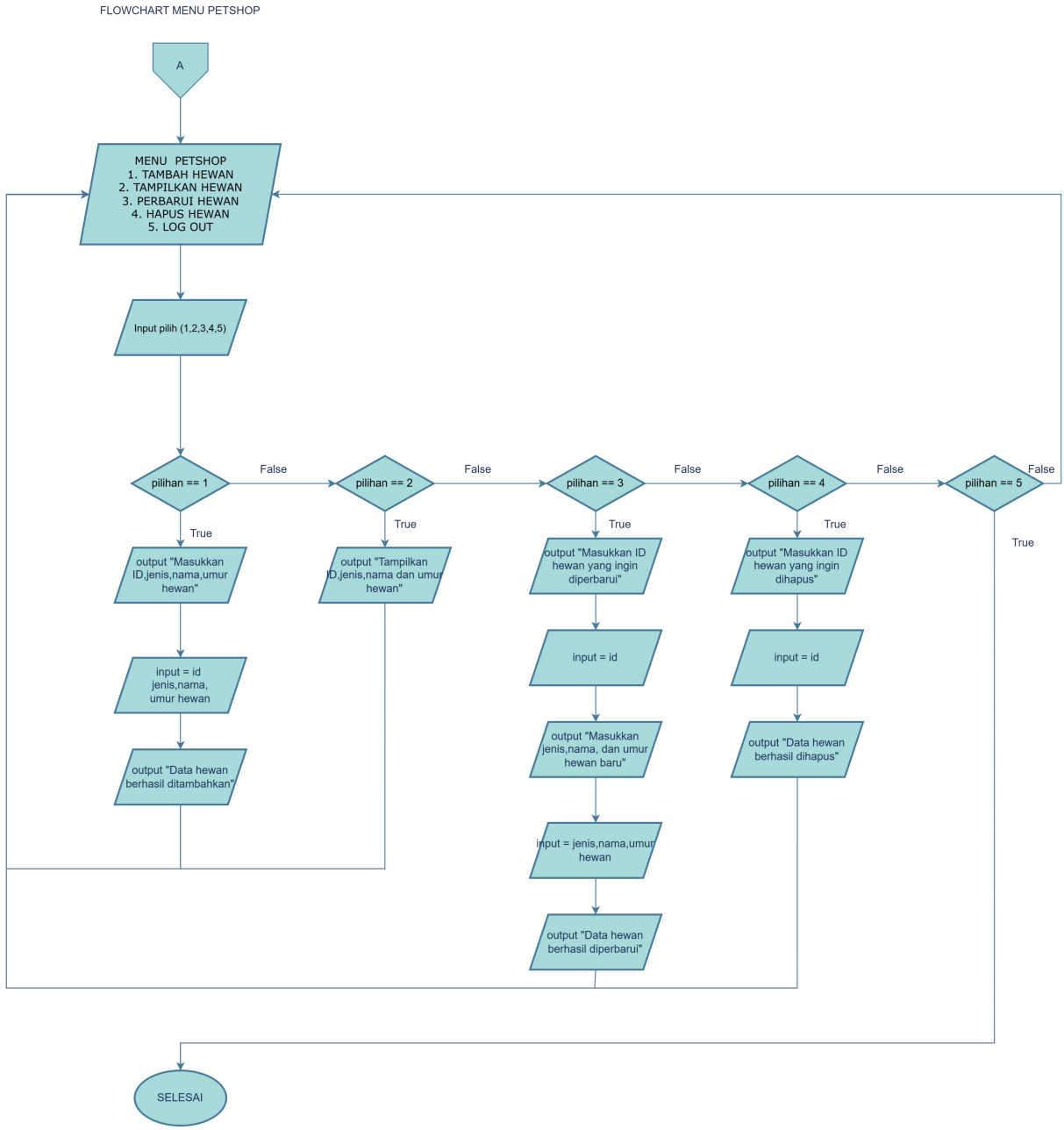
SAMARINDA

2025

1. Flowchart



Gambar 1.1 Flowchart Login



Gambar 1.2 Flowchart Menu Petshop

2. Analisis Program

Program Manajemen Petshop ini bertujuan untuk membantu dalam mengelola data hewan secara efisien. Dengan fitur CRUD. Program ini dapat menambahkan,melihat,memperbarui, dan menghapus data hewan dengan mudah. Lalu manfaat dari program ini yaitu meningkatkan efisiensi dalam pencatatan data hewan dan memudahkan pencarian dan pembaruan data hewan. Sebenarnya program ini sama seperti posttest 2 yang kemarin tetapi disini ada perubahan pada 3 variabel array satu dimensi menjadi 1 variable array of struct.

3. Source Code

A. Pendefinisian Struktur dan Variabel Global

```
const int limit_hewan = 100;

struct Hewan
{
    string id;
    string jenis;
    string nama;
    int umur;
};

Hewan daftar_hewan[limit_hewan];
int jumlah_hewan = 0;
```

Program ini menentukan batas maksimal jumlah hewan yang bisa disimpan (100) dengan menggunakan struct Hewan yang memiliki atribut id, jenis, nama, dan umur, lalu menyimpan daftar hewan dalam array Hewan daftar_hewan[limit_hewan]; serta melacak jumlah hewan yang sudah dimasukkan dengan variabel int jumlah_hewan = 0;.

B. Login

```
int main()
{
    string Nama, NIM;
    bool login = false;
    int percobaan_login = 0;

    while (percobaan_login < 3 && !login)
    {
        cout << "Masukkan Nama : ";
        cin >> Nama;
        cout << "Masukkan NIM : ";
        cin >> NIM;

        if (Nama == "bila" && NIM == "041")
        {
            login = true;
            cout << "Login berhasil, Welcome " << Nama << "!\n";
        }
        else
        {
            percobaan_login++;
            cout << "Nama atau NIM salah, Silahkan coba lagi!\n";
        }
    }
    if (!login)
    {
        cout << "Terlalu banyak percobaan login, program dihentikan !\n";
        return 0;
    }
```

Variabel login terdiri dari Nama dan NIM untuk menyimpan input user, bool login = false; sebagai status login awal, serta int percobaan_login = 0; untuk menghitung jumlah percobaan login, dengan proses login menggunakan looping while yang membatasi maksimal 3 kali percobaan, di mana jika user memasukkan Nama "bila" dan NIM "041", login berhasil, tetapi jika salah, percobaan_login bertambah, dan jika percobaan habis (>=3), program dihentikan dengan return 0;.

C. Menu Petshop

```
while (true)
    {
        cout << "\n======================";
        cout << "\n|    MENU PET SHOP    |";
        cout << "\n======================";
        cout << "\n|  1. TAMBAH HEWAN    |";
        cout << "\n|  2. TAMPILKAN HEWAN |";
        cout << "\n|  3. UBAH HEWAN      |";
        cout << "\n|  4. HAPUS HEWAN     |";
        cout << "\n|  5. KELUAR PROGRAM  |";
        cout << "\n======================";
        cout << "\nPilih (1/2/3/4/5): ";
        int pilihan;
        cin >> pilihan;
```

Ini adalah tampilan menu petshop.

D. Fitur Menambahkan Data Hewan

```
if (pilihan == 1)
        {
            if (jumlah_hewan < limit_hewan)
            {
                cout << "Masukkan ID Hewan: ";
                cin >> daftar_hewan[jumlah_hewan].id;
                cout << "Masukkan jenis hewan: ";
                cin >> daftar_hewan[jumlah_hewan].jenis;
                cout << "Masukkan Nama Hewan: ";
                cin >> daftar_hewan[jumlah_hewan].nama;
                cout << "Masukkan Umur Hewan: ";
                cin >> daftar_hewan[jumlah_hewan].umur;
                jumlah_hewan++;
                cout << "Data hewan telah berhasil ditambahkan.\n";
            }
            else
            {
                cout << "Maaf, tidak dapat menambahkan hewan lagi\n";
            }
        }
```

Program mengecek apakah jumlah hewan belum mencapai batas (limit_hewan), lalu memungkinkan pengguna memasukkan data hewan (ID, jenis, nama, umur), di mana jika berhasil jumlah_hewan bertambah, tetapi jika sudah penuh, program menampilkan pesan error, dengan data hewan disimpan dalam daftar_hewan, yaitu sebuah array dari struct Hewan yang setiap elemennya berisi informasi tentang satu hewan, termasuk ID, jenis, nama, dan umur, serta memiliki batas maksimal limit_hewan sebesar 100.

E. Fitur Menampilkan Data Hewan

```
else if (pilihan == 2)
        {
            if (jumlah_hewan == 0)
            {
                cout << "Tidak ada data hewan.\n";
            }
            else
            {
                cout << "\n========= DATA HEWAN =========\n";
                cout << "ID\tJenis\tNama\tUmur\n";
                cout << "-----------------------------------\n";
                for (int i = 0; i < jumlah_hewan; i++)
                {
                    cout << daftar_hewan[i].id << "\t"
                         << daftar_hewan[i].jenis << "\t"
                         << daftar_hewan[i].nama << "\t"
                         << daftar_hewan[i].umur << " tahun\n";
                }
                cout << "-----------------------------------\n";
            }
        }
```

Jika tidak ada hewan, program menampilkan pesan "Tidak ada data hewan", tetapi jika ada, program menampilkan semua data hewan termasuk ID, jenis, nama, dan umur.

F. Fitur Memperbarui Data Hewan

```
else if (pilihan == 3)
        {
            string id;
            cout << "Masukkan ID hewan yang ingin diubah: ";
            cin >> id;
            bool terverifikasi = false;
            for (int i = 0; i < jumlah_hewan; i++)
            {
                if (daftar_hewan[i].id == id)
                {
                    cout << "Masukkan jenis baru: ";
                    cin >> daftar_hewan[i].jenis;
                    cout << "Masukkan nama baru: ";
                    cin >> daftar_hewan[i].nama;
                    cout << "Masukkan umur baru hewan: ";
                    cin >> daftar_hewan[i].umur;
                    terverifikasi = true;
                    cout << "Data hewan berhasil diperbarui.\n";
                    break;
                }
            }
            if (!terverifikasi)
                cout << "Hewan tidak ditemukan!\n";
        }
```

Program mencari hewan berdasarkan ID, jika ditemukan pengguna dapat mengubah datanya, tetapi jika tidak ditemukan, program menampilkan pesan "Hewan tidak ditemukan".

G. Fitur Menghapus Data Hewan

```
else if (pilihan == 4)
        {
            string id;
            cout << "Masukkan ID hewan yang ingin dihapus: ";
            cin >> id;
            bool terverifikasi = false;
            for (int i = 0; i < jumlah_hewan; i++)
            {
                if (daftar_hewan[i].id == id)
                {
                    for (int j = i; j < jumlah_hewan - 1; j++)
                    {
                        daftar_hewan[j] = daftar_hewan[j + 1];
                    }
                    jumlah_hewan--;
                    terverifikasi = true;
                    cout << "Data hewan berhasil dihapus.\n";
                    break;
                }
            }
            if (!terverifikasi)
                cout << "Hewan tidak ditemukan!\n";
        }
```

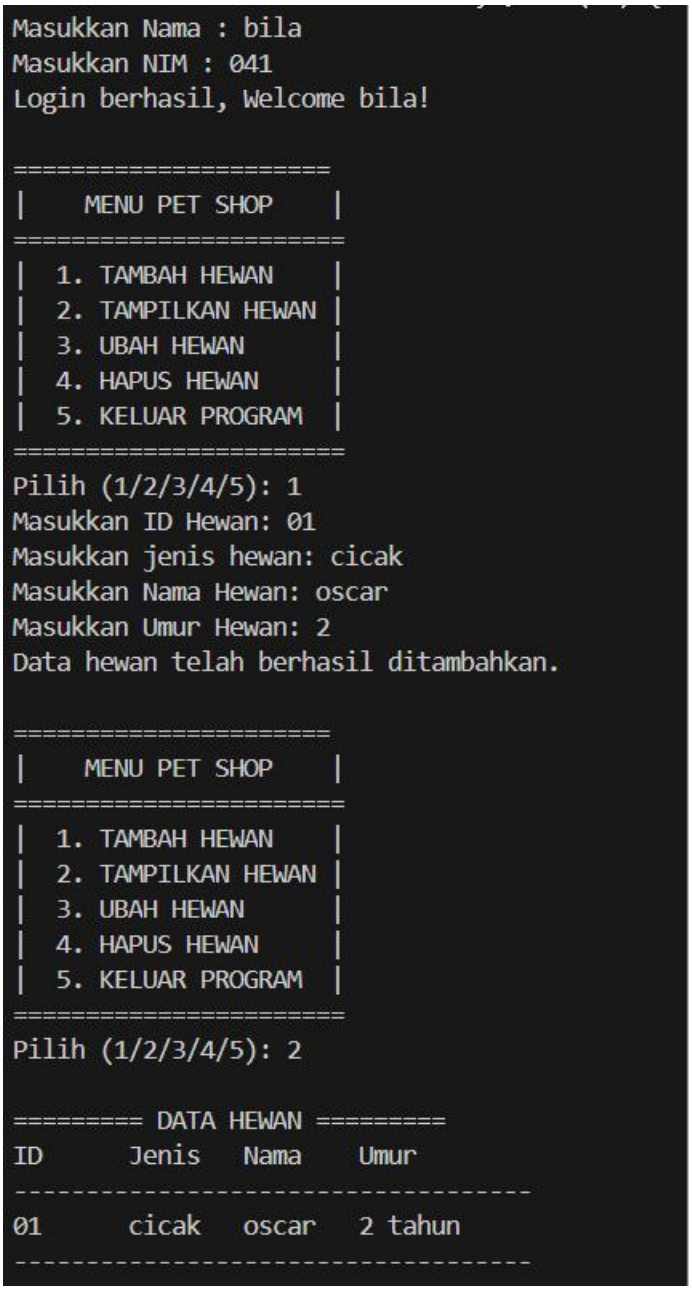
Program mencari hewan berdasarkan ID, jika ditemukan elemen array digeser untuk menghapusnya, tetapi jika tidak ditemukan, program menampilkan pesan error.

H. Fitur Keluar Dari Program

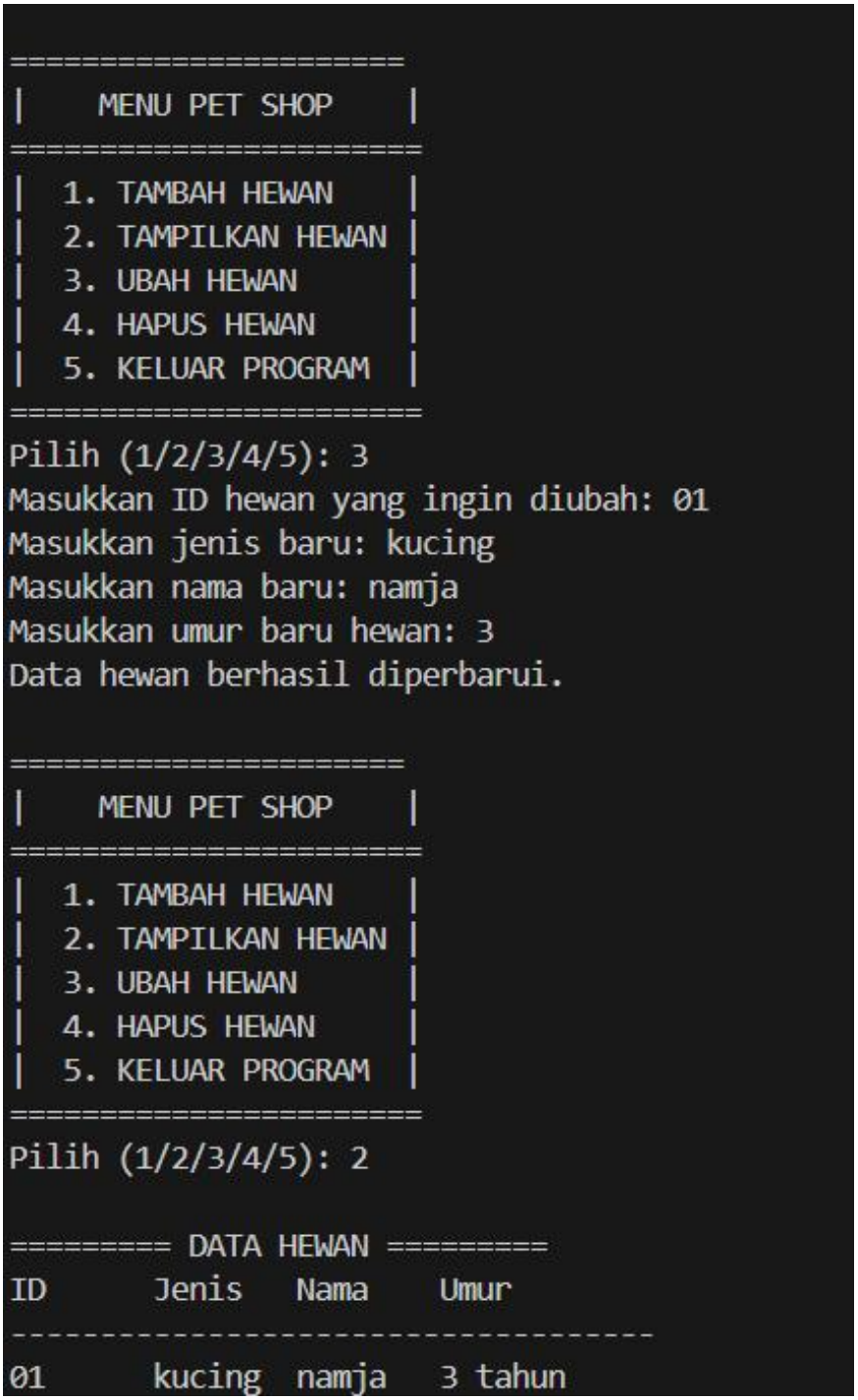
```
 else if (pilihan == 5)
        {
            cout << "Terima kasih telah menggunakan program petshop ini.\n";
            return 0;
        }
        else
        {
            cout << "Pilihan tidak valid.\n";
        }
    }
    return 0;
}
```

Program akan berakhir ketika pengguna memilih opsi keluar, di mana sistem menampilkan pesan terima kasih dan menghentikan eksekusi dengan perintah return 0, memastikan bahwa tidak ada proses lebih lanjut yang berjalan setelahnya.

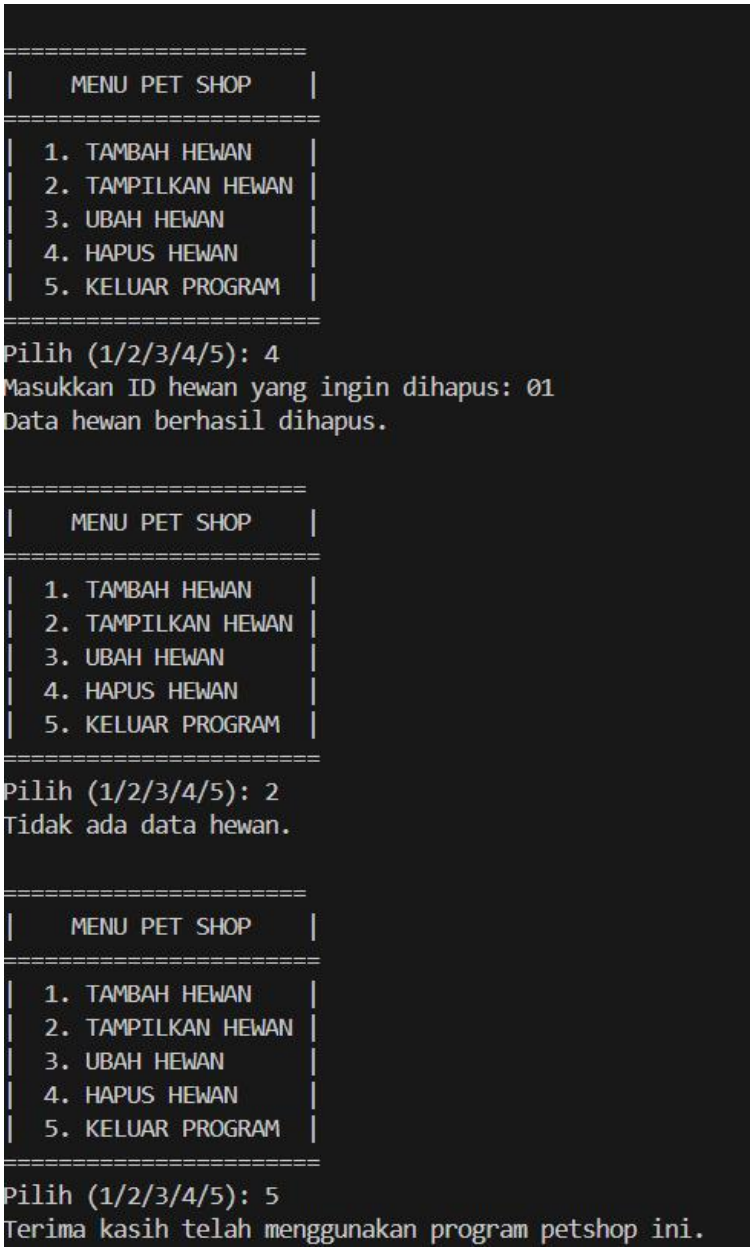
4. Hasil Output



Gambar 4.1 Tampilan Terminal



Gambar 4.2 Tampilan Terminal



Gambar 4.3 Tampilan Terminal

5. Langkah-Langkah Git pada VSCode

```
PS C:\Users\MSI GF63\Documents\Praktikum apl\praktikum-apl> git add .
PS C:\Users\MSI GF63\Documents\Praktikum apl\praktikum-apl> git commit -m "Finish Post test 3"
[main f46b0b2] Finish Post test 3
 4 files changed, 54 insertions(+)
 create mode 100644 .vscode/settings.json
 create mode 100644 kelas/pertemuan-3.cpp
 create mode 100644 kelas/pertemuan-3.exe
 create mode 100644 post-test/post-test-3/2409106041-NabilaPutriKarni-PT-3.pdf
PS C:\Users\MSI GF63\Documents\Praktikum apl\praktikum-apl> git push
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 16 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (16/16), 888.54 KiB | 21.16 MiB/s, done.
Total 16 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/lymoonty/praktikum-apl.git
   d5fcc7f..f46b0b2  main -> main
PS C:\Users\MSI GF63\Documents\Praktikum apl\praktikum-apl>
```

Gambar 5.1

Git add berfungsi untuk menambahkan perubahan pada file atau seluruh proyek ke dalam staging area, sehingga perubahan tersebut siap untuk disimpan dalam commit berikutnya.

Git commit digunakan untuk menyimpan perubahan yang telah ditambahkan ke staging area secara permanen ke dalam repository lokal dengan menyertakan pesan deskriptif sebagai catatan perubahan.

Git push digunakan untuk mengunggah commit dari repository lokal ke repository online, memungkinkan kode sumber dapat diakses dari mana saja atau dibagikan dengan orang lain.