Deskripsi Desain Program JuiceShop
Ahmad Izzuddin Azzam, Quiz 2 DPBO

Tujuan Program:

Pada quiz 2 ini, kami diarahkan utnuk latihan friend, exception, dan template pada class maupun method yang ada, dan tema yang diambil kali ini adalah sistem penjualan jus dibeberapa toko jus yang menggunakan berbagai jenis buah sebagai bahan dasarnya,rogram ini menangani pengelolaan stok buah, pemesanan minuman oleh pelanggan, dan laporan penjualan dan ada beberapa exception disini yang dibuat, seperti *Stock Exception*, *Order Exception*, *Price Exception*, dan *InvalidFormatException*.

Struktur Program:

Kelas-kelas yang digunakan:

1. Fruit (Class Template):
   Disini class Fruit mempunyai atribut berupa name dengan tipe data string, dan price_per_kg dengan tipe data fleksible (menggunakan Template yang saya kasih nama J), kenapa, karena bisa aja harga itu nggk double atau float, melainkan integer jadi nggak ada komanya, dan di Fruit ini ada exception harga, dimana harga yang diinputkan oleh user tidak boleh mines, alias harus postitif (menggunakan Price Exception).

2. Supplier:
   Class supplier ini mempunyai atribut stock, dimana ini adalah sebuah vector (atau memakai composite lah intinyamah) dari fruit, yang fruit ini adalah tipe class template, ini mengelola stok buah dan menyediakan metode untuk menambahkan buah ke stock serta mencari buah berdasarkan nama

3. Drink (Template Class):
   Class drink ini memiliki atribut ingredient atau bahan dasar dari siminuman jus inih, memakai buah apasi yang tipe datanya fleksible sebenernya, makanya make template, dan mempunyai volume tertentu.
   Menyimpan informasi tentang buah yang digunakan dan volume yang dipesan,
   Dapat memeriksa apakah volume yang diminta melebihi batas yang ditentukan atau tidak, alias tidak bocor minta lebih dari yang sipenjual buatin, misal segela cmn bisa seliter, dia minta 3 liter, dan ini ditangguhkan dengan menggunakan OrderException.

4. Customer:
   customer ini ada name aja ya, menggunakan string seperti biasa, dan ada geter setternya, dan dia bisa mengakses Atribut yang ada di Juice Shop, karena dia ada fungsi orderDrink yang membutuhkan informasi atribut dari jus yang dia beli, dari namanya juga volumenya, nahh disini kita menggunakan sistem Friend

5. JuiceShop:
   Juice shop ini mempunyai list untuk buah apa ajasih yang dijual disini, dan customernya tu ada siapa aja, yang kita simpan dalam tipe data vector, btw vector itu juga termasuk template yaa, karena dia ada vector< >, bisa diisi apa aja, nah kita juga ada tipe data map untuk menyimpan sales yang ada dijuice shop tersebut. dan ada method add customer, add fruot, dan proses ordernya

Desain Alur Program:

```
int main() {
    Supplier supplier;
    supplier.addFruitToStock("Apel", 10.0);
    supplier.addFruitToStock("Jeruk", 9.5);
    supplier.addFruitToStock("Mangga", 12.0);
    supplier.addFruitToStock("Semangka", 7.0);
    supplier.addFruitToStock("Nanas", 8.0);
    supplier.addFruitToStock("Stroberi", 15.0);

    cout << "\n=== SISTEM MULTI JUICE SHOP DIMULAI ===\n" << endl;
```

Program ini dimulai dengan inisialisasi objek Supplier yang menambahkan buah-buah ke dalam stok si supliernya.

1.
```
// ================ JuiceShop 1 ================
JuiceShop shopA;
shopA.addFruitToShop(supplier.findFruitByName("Apel"));
shopA.addFruitToShop(supplier.findFruitByName("Jeruk"));

Customer custA1("Ali");
Customer custA2("Budi");
shopA.addCustomer(custA1);
shopA.addCustomer(custA2);

Drink<Fruit<double>> drinkA1(supplier.findFruitByName("Apel"), 2.0);
Drink<Fruit<double>> drinkA2(supplier.findFruitByName("Jeruk"), 3.0);

cout << "\n[ Juice Shop A - Pemesanan ]" << endl;
try {
    shopA.processOrder(custA1, drinkA1);
    shopA.processOrder(custA2, drinkA2);
} catch (exception& e) {
    cerr << "Exception: " << e.what() << endl;
}
```

```
JuiceShop shopB;
shopB.addFruitToShop(supplier.findFruitByName("Mangga"));
shopB.addFruitToShop(supplier.findFruitByName("Semangka"));

Customer custB1("Citra");
Customer custB2("Dina");
shopB.addCustomer(custB1);
shopB.addCustomer(custB2);

Drink<Fruit<double>> drinkB1(supplier.findFruitByName("Mangga"), 1.5);
Drink<Fruit<double>> drinkB2(supplier.findFruitByName("Semangka"), 2.5);

cout << "\n[ Juice Shop B - Pemesanan ]" << endl;
try {
    shopB.processOrder(custB1, drinkB1);
    shopB.processOrder(custB2, drinkB2);
} catch (exception& e) {
    cerr << "Exception: " << e.what() << endl;
}
```

```
// ================= JuiceShop 3 =================
JuiceShop shopC;
shopC.addFruitToShop(supplier.findFruitByName("Nanas"));
shopC.addFruitToShop(supplier.findFruitByName("Stroberi"));

Customer custC1("Eka");
Customer custC2("Fajar");
shopC.addCustomer(custC1);
shopC.addCustomer(custC2);

Drink<Fruit<double>> drinkC1(supplier.findFruitByName("Nanas"), 1.0);
Drink<Fruit<double>> drinkC2(supplier.findFruitByName("Stroberi"), 3.0);
Drink<Fruit<double>> overVol(supplier.findFruitByName("Stroberi"), 20.0); // Test OrderException

cout << "\n[ Juice Shop C - Pemesanan ]" << endl;
try {
    shopC.processOrder(custC1, drinkC1);
    shopC.processOrder(custC2, drinkC2);
    shopC.processOrder(custC1, overVol); // akan melempar exception
} catch (StockException& e) {
    cerr << "Stock Exception: " << e.what() << endl;
} catch (OrderException& e) {
    cerr << "Order Exception: " << e.what() << endl;
} catch (PriceException& e) {
    cerr << "Price Exception: " << e.what() << endl;
} catch (InvalidFruitException& e) {
    cerr << "Invalid Fruit Exception: " << e.what() << endl;
} catch (exception& e) {
    cerr << "General Exception: " << e.what() << endl;
}
```

Beberapa toko jus (JuiceShop) dibuat dan diisi dengan buah dari stok supplier.

2. Toko jus kemudian menerima pelanggan, penlanggannya memesan, dan akhirnya memproses pemesanan minuman.

3.
```
try {
    shopC.processOrder(custC1, drinkC1);
    shopC.processOrder(custC2, drinkC2);
    shopC.processOrder(custC1, overVol); // akan melempar exception
} catch (StockException& e) {
    cerr << "Stock Exception: " << e.what() << endl;
} catch (OrderException& e) {
    cerr << "Order Exception: " << e.what() << endl;
} catch (PriceException& e) {
    cerr << "Price Exception: " << e.what() << endl;
} catch (InvalidFruitException& e) {
    cerr << "Invalid Fruit Exception: " << e.what() << endl;
} catch (exception& e) {
    cerr << "General Exception: " << e.what() << endl;
}
```

Program menangani berbagai exception seperti jika harga buah negatif, volume pesanan terlalu besar, atau buah yang tidak tersedia di toko, yang sudah diatur di folder exceptions

4.
```
cout << "\n=== LAPORAN PENJUALAN ===\n" << endl;
cout << "[ Shop A ]" << endl;
shopA.displaySales();

cout << "\n[ Shop B ]" << endl;
shopB.displaySales();

cout << "\n[ Shop C ]" << endl;
shopC.displaySales();

cout << "\n=== APLIKASINYA SUDAH SELESAI ===" << endl;
return 0;
```

Laporan penjualan ditampilkan setelah seluruh pemesanan diproses, menunjukkan total penjualan per toko pada akhir dari programnya.

Exception Handling:

1. Stock Exception:
   Akan Dilempar ketika buah yang dicari tidak tersedia dalam stok.

2. Order Exception:
```
Drink<Fruit<double>> overVol(supplier.findFruitByName("Stroberi"), 20.0); // Test OrderException
```
```
Order Exception: Volume pesanan melebihi batas yang diizinkan!
```
   Akan Dilempar jika volume pesanan melebihi kapasitas yang diizinkan, jadi gabisa mesen minuman 2 liter yang maksimalnya 1 liter.

3. Price Exception:
```
try
{
    Drink<Fruit<double>> overVol(supplier.findFruitByName("Stroberi Mines Harga"), -100); // Price Exception

}
catch(PriceException& e)
{
    cerr << "Price Exception: " << e.what() << endl;
}
```
```
terminate called after throwing an instance of 'InvalidFruitException'
  what():  Buah tidak valid!
```

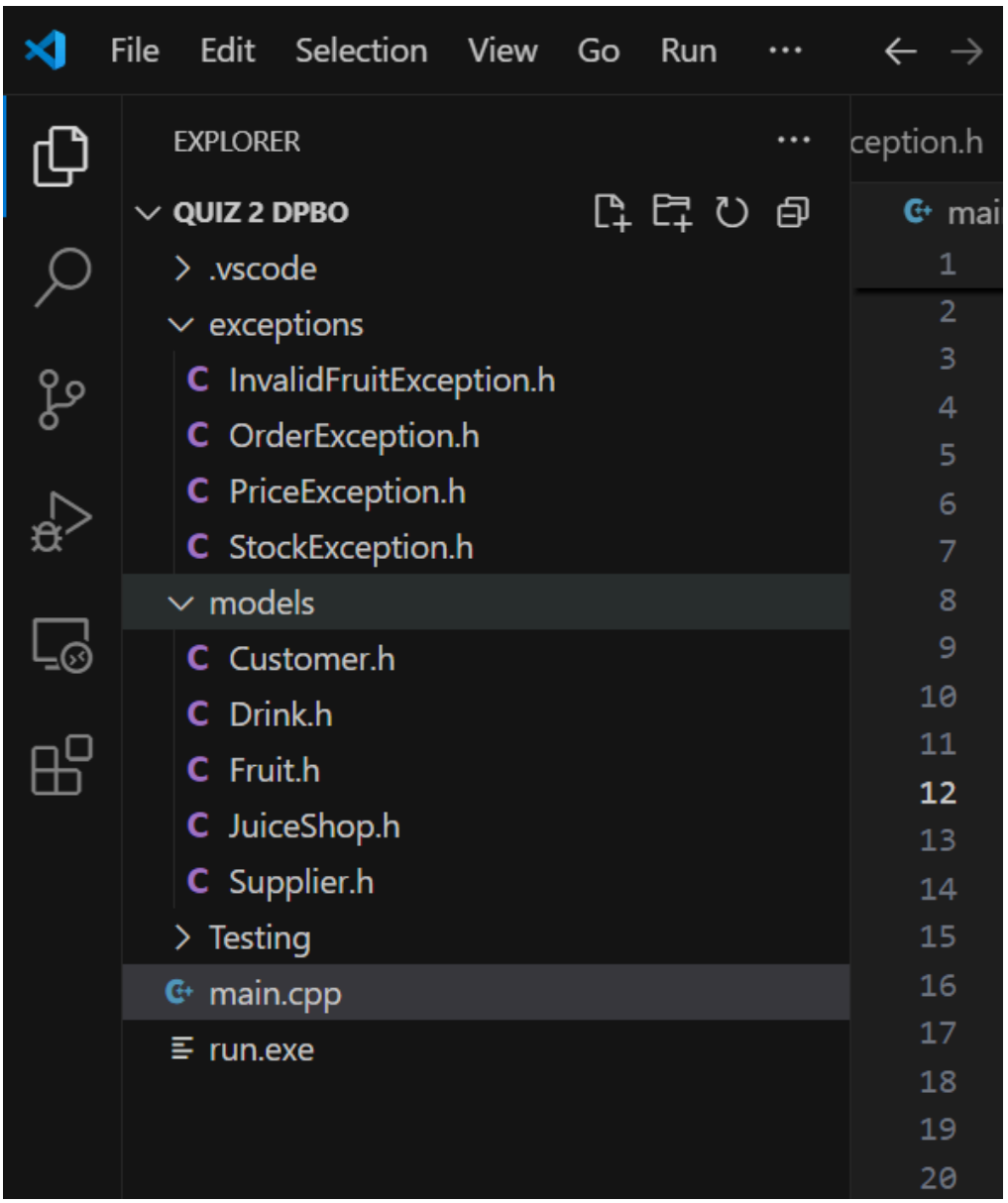
Akan Dilempar jika harga buah negatif, harus positif intinya.

4. Invalid Fruit Exception:
   Akan Dilempar jika jenis buah tidak ditemukan di toko jusyang customer akan beli.

Penggunaan Friend:

1. friend class JuiceShop; pada class Customer
   karena ada fungsi untuk sicustomernya beli buah apa aja, yang berarti perlu untuk mengambil atribut yang ada pada class customernya

Struktur Direktori:



Penjelasan Alur Eksekusi Program:

1. Inisialisasi Supplier:
    a. Menambahkan buah ke dalam stok.
    b. Menambahkan beberapa buah ke beberapa toko jus.

2. Proses Pemesanan:

    a. Toko jus menerima pesanan dari pelanggan.
    b. Jika terjadi kesalahan dalam pemesanan, program akan menampilkan pesan error sesuai dengan jenis exception yang terjadi.

3. Laporan Penjualan:
    a. Setelah semua pemesanan selesai, laporan penjualan ditampilkan per toko.

4. Pentingnya Menggunakan Template:
    a. Program ini menggunakan template untuk kelas Fruit dan Drink agar dapat bekerja dengan berbagai tipe data, seperti int dan double, untuk harga dan volume. Ini membuat program lebih fleksibel dan dapat menangani berbagai jenis data tanpa mengubah logika dasar.

5. Pengujian:
    a. Program diuji untuk memastikan bahwa:
    b. Setiap jenis exception dapat ditangani dengan benar.
    c. Pemesanan dapat diproses dengan volume yang sesuai dan buah yang tersedia.
    d. Laporan penjualan dihasilkan dengan benar setelah semua transaksi.