**2022 Spring OOP Assignment Report**

과제 번호 : prob2\_2

학번 : 20210273

이름 : 하태혁

Povis ID : hth021002

**명예서약 (Honor Code)**

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.

I completed this programming task without the improper help of others.

프로그램을 하다 보면 결정해야 할 세부 사항이 많은데, 이러한 세부 사항을 처리한 방법과 이유를 보고서에 쓰십시오.

문제가 여러 개인 경우, 각 문제별로 정리해서 작성합니다.

각 문항별 설명은 편의를 위한 것으로, 삭제하고 제출한다.

1. **프로그램 개요**

이 프로그램은 A구역과 B구역의 손님 수를 바탕으로 피자 가격과 배달료를 계산한 뒤 피자를 배송하여 피자 가게의 수입과 라이더 들의 수입, 배달 시간, 구역 손님들의 돈과 시간을 계산하여 출력하는 프로그램이다.

Pizza클래스와 Customor클래스, Rider클래스, Vector\_Customer & Vector\_Rider class 를 구현하였다. 각 클래스의 헤더파일마다 함수의 정의가 포함되어있는 cpp파일을 제작하였고 #pragma once 를 사용하여 main.cpp에서 같은 헤더파일이 두 번 선언되어 발생하는 에러를 막고자 하였다. Main.cpp에서는 아래 사진에 나와 있는 바와 같이 5개의 헤더파일과 <iosteam>을 선언하여 사용한다. 그러므로 prob2는 아래 5개에 해당한 클래스를 제작한 것이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. **프로그램의 구조 및 알고리즘**
2. Pizza class

클래스의 멤버변수와 메소드의 기본적인 설명은 첨부되어 있는 보고서와 같다. 문제 공통 규칙에 설명되어있는 바와 같이 index기준 선입선출의 원리를 적용하여 클래스를 구현하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

각 구역 고객의 수는 0명 이하 10명 이상이 될 수 없고 각 구역 고객 수의 합은 항상 10을 유지해야 하므로 각 구역 고객수를 9명 미만이 되는 경우에만 타지역 고객을 선입선출 원리에 따라 배출하는 식으로 구성하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

deliver()의 경우 A구역부터 배달을 완료해야 하고 Rider는 5명이 순서대로 각 구역의 고객을 index 순으로 방문해야 하므로 for문을 사용하였고 int형 변수 i와 j를 사용하여 각구역 고객을 index 0부터 방문하면서 Rider는 I변수 하나만 사용하여 반복문을 진행하면서 라이더 배치 원칙을 따르고자 하였다.

1. Customer class & Rider class

멤버변수와 멤버함수에 대한 설명은 설명서에 제시된 설명의 기능을 구현하는 식으로 제작하였다. 텍스트이(가) 표시된 사진

자동 생성된 설명

다음과 같이 함수의 인자와 멤버함수의 변수 이름이 같은 경우 멤버변수임을 나타내기 위해 this를 사용하여 구현하였다. <<operator를 구현하면서는 friend 키워드를 사용하여 외부 함수가 클래스의 멤버변수에 접근할 수 있도록 구현하였다.

1. Vector\_Customer & Vector\_Rider class

기본적인 멤버변수와 멤버함수의 구현은 기존 prob1에서 구현한 Vector 클래스의 기능과 거의 동일하며 float형이던 함수를 object형으로 바꿔 사용하였다.

1. **토론 및 개선**

Prob2에서 구현한 Vector\_Customer & Vector\_Rider class 에서 만약 template를 사용했다면 Object형 뿐만 아니라 다른 type에 대해서도 활용이 가능하므로 generic programming을 해볼 수 있었을 것으로 기대한다. 그러나 prob1에서 제작한 Vector 클래스의 경우 일반적인 벡터 구현을 하는데 사용할 수 있으므로 template를 사용했을 때 int나 char등의 다른 데이터 타입에 대해서도 사용할 수 있다는 명확한 장점이 있지만, prob2에서 사용한 Vector\_[clacc] 클래스의 경우에는 손님이나 라이더의 정보를 저장하기 위해서는 반드시 object형으로 구현해야 했기에 template으로 사용한다고 해서 prob1만큼의 큰 장점이 있을 것이라고는 예상되지 않는다.

1. **참고 문헌**

설명문과 기존에 구현한 prob1을 제외하고는 아무것도 참고하지 않았다.