**Jakub Jakubowski (259544)**

Dokumentacja projektu

**Sposób działania programu:**

Aplikacja „sklep” jest symulacją prowadzenia prawdziwego sklepu. Jej zadanie w sposób rzeczywisty jest w stanie ułatwić prowadzenie magazynu każdego sklepu spożywczego.   
 Początkowe zadanie aplikacji to rozróżnienie osoby, która aktualnie z niej korzysta. Tutaj program jest w stanie nas przenieść do interfejsu klienta, pracownika, lub w przeciwnym wypadku, jeśli nie rozpozna w nas żadnej z poprzednich osób, poprosi nas o opuszczenie sklepu i zakończy dla nas działanie aplikacji.  
 Rozpatrując pierwszy przypadek jako „klient” jesteśmy w stanie przechodzić między kategoriami i wkładać do koszyka dowolne produkty. Zawsze jest widoczna liczba rzeczy w koszyku jak i suma jaką przyjdzie nam za to zapłacić. Aplikacja kończy swoje działanie, jeśli użytkownik przejdzie do kasy, tam zostanie również przedstawiona łączna suma do zapłaty jak i ilość kupionych produktów. W tym momencie również, zakupione produkty są wydawane z magazynu (plik „magazyn.txt”).  
 Natomiast podając się za „pracownika” interfejs nie będzie bardzo skomplikowany. Zostają wypisane wszystkie produkty, a obok nich znajdujący się indywidualny kod. Podanie tego kodu spowoduje dodanie tego produktu do magazynu, który zapisuje się w pliku „magazyn.txt”.

**Struktura oprogramowania:**

Działanie programu głównie opiera się na połączonych ze sobą „switch’ach” (instrukcje switch). W nich zawarte są funkcje void, których zadanie jest wyświetlanie interfejsu w zależności, gdzie się znajdujemy np. jesteśmy klientem to wyświetl interfejs klienta itp.   
 Wszystkie elementy obiektowe w tym programie opierają się na 3 podstawowych klasach (pieczywo, nabiał, alkohole). Każda z wymienionych klas posiada swoje klasy którym dziedziczy np. ilość.  
 W klasie „pieczywo” można zaobserwować takie elementy obiektowe jak: Konstruktor, dziedziczenie konstruktorów, funkcja wirtualna, dziedziczenie public elementów publicznych i protected, wirtualny destruktor. W głównym programie tworzymy tablice wskaźników klasy abstrakcyjnej, za pomocą której będziemy operować na klasach dziedziczących takich jak: tostowy, bulka, paczek. Wraz z ich tworzeniem nadajemy im wartość ceny w nawiasie obok nazwy klasy. Dodatkowo przy tworzeniu „obiektów” zastosowałem operator do dynamicznej alokacji pamięci „new” i „delete”. Zadaniem funkcja wirtualnej cena1() jest jedynie zwracanie ceny danego obiektu. Stworzenie jej jest bardzo przydatne jeśli chcemy dostać się do wartości ceny która jest chroniona (protected), innym wyjściem mogłoby być stworzenie funkcji zaprzyjaźnionej, lub metody w klasie.   
 W klasie „nabial” wyjątkowo nie korzystałem z pliku .cpp, ponieważ nie zawiera wiele danych, więc w tym przypadku pozwoliłem sobie na prace jedynie w pliku nagłówkowym „nabial.h”. Z elementów obiektowych zastosowałem funkcje zaprzyjaźnioną, jednak jedynie do klasy „jajka”. Jej zadanie nie jest skomplikowane, jedynie informuje kupca, ile zostało sztuk w magazynie po dokonaniu zakupu właśnie jaj kurzych. Ceny danych produktów (obiektów) są już dane z góry przy ich tworzeniu.  
 Klasa „alkohole” wyróżnia się od poprzedniej klasy tym, że posiada ona konstruktory do każdej klasy dziedziczącej z klasy alkohole, w których cena w przypadku nie nadania jej przy tworzeniu obiektu, nadała je sama.

