

Sprawozdanie z zadania projektowego

Treść zadania: Projekt i implementacja sklepu internetowego.

Funkcjonalność aplikacji.

a) Opis rozpatrywanej dziedziny:

Sklep internetowy to serwis dający możliwość kupowania produktów za pośrednictwem Internetu. Stworzenie systemów sklepów internetowych dało możliwość uruchomienia handlu internetowego. Popularne systemy sklepów internetowych to np. osCommerce lub Quick.Cart oparte na licencjach GPL. Towary tak jak na półkach w hipermarkecie prezentowane są na stronie internetowej sklepu internetowego.

Kupowanie produktów w sklepie internetowym odbywa się głównie w kilku najważniejszych krokach:

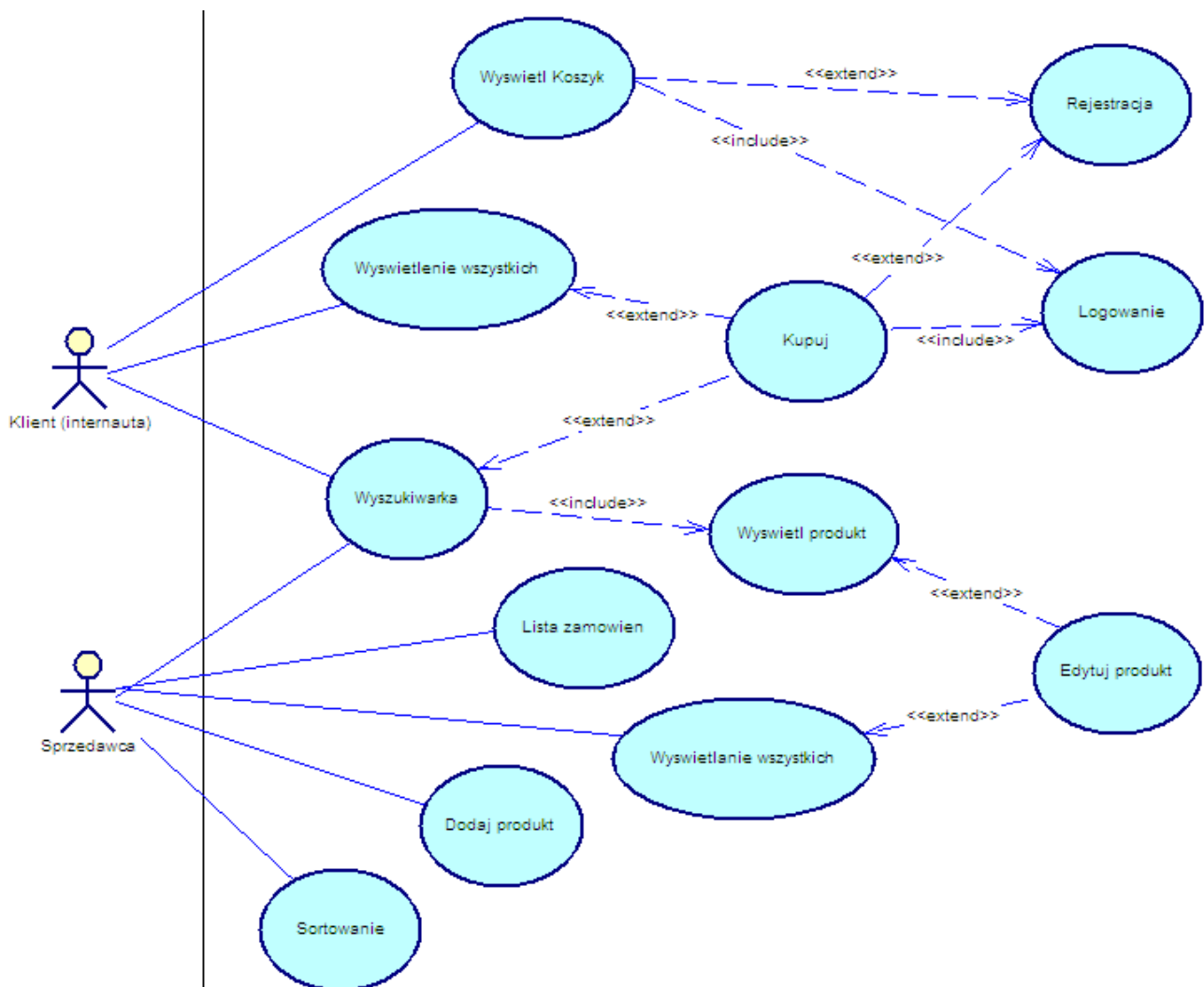
- **Krok pierwszy** - Klient wyszukuje w sklepie pożądaną towar. Może to zrobić za pomocą wyszukiwarki lub w większych sklepach internetowych poprzez przeglądanie kategorii produktów, na które zostały podzielone. Krok ten kończy się, gdy klient doda wszystkie pożądane przez niego produkty do swojego wirtualnego koszyka (niektóre proste sklepy internetowe nie posiadają koszyka, tylko od razu wyświetlają formularz zamówienia).
- **Krok drugi** - Klient w tym kroku składa zamówienie na towar znajdujący się w koszyku. W większości przypadków system żąda od klienta zalogowania się lub, jeśli klient jest pierwszy raz w danym sklepie rejestracji klienta w bazie. Podczas rejestracji klient wypełnia dane teleadresowe oraz dane kontaktowe.
- **Krok trzeci** - Zapłata za towar. W zależności od sklepu internetowego można tego dokonać za pomocą przelewu, kartą kredytową lub zapłacić przy odbiorze.
- **Inne kroki** - Telefoniczne lub e-mailowe potwierdzenie zamówienia.

b) Założenia, co do funkcjonalności aplikacji:

Stworzona przeze mnie aplikacja sklepu internetowego z faktu, iż została napisana w języku C++ jest jedynie symulatorem prawdziwych systemów sklepów internetowych napisanych w językach np. PHP z wykorzystaniem bazy danych np. MySQL. Założeniem moim było, aby zaprojektować w aplikacji wszystkie podstawowe funkcje sklepu internetowego:

- funkcje dostępne dla klienta: wyświetlanie wszystkich produktów, wyszukiwanie produktów, kupowanie, rejestracja, logowanie, koszyk (lista zamówień)
- funkcje dostępne z poziomu administratora: wyszukiwarka, wyświetlanie produktów, dodawanie produktów, sortowanie, lista zamówień klientów, edycja produktów w bazie, logowanie do panelu administratora.
- funkcje bazy danych spełniają pliki: magazyn.txt (baza produktów), users.txt (baza klientów), zamowienia.txt (baza zamówień).

Model przypadków użycia (Use Case Diagram)



Scenariusze użycia dla klienta.

- 1 Wyświetl koszyk
 - 1.1 wybierz tryb klienta,
 - 1.2 wybierz koszyk,
 - 1.3 zaloguj się,
 - 1.3.1 jeśli nie masz konta idź do punktu 1.4,
 - 1.3.2 jeśli udane idź do punktu 1.5,
 - 1.3.3 jeśli nieudane wróć do punktu 1.3,
 - 1.4 zarejestruj się,
 - 1.5 wyświetl koszyk,
 - 1.6 przeglądaj zamówione produkty.
- 2 Wyświetl wszystkie produkty
 - 2.1 wybierz tryb klienta,
 - 2.2 wyświetl wszystkie produkty,
 - 2.2.1 jeśli istnieją idź do punktu 2.3
 - 2.2.2 jeśli brak produktów w bazie idź do punktu 2.2
 - 2.3 przeglądaj produkty.

- 3 Wyszukaj produkty
 - 3.1 wybierz tryb klienta,
 - 3.2 wybierz wyszukiwarke,
 - 3.3 wybierz kryteria wyszukiwania
 - 3.4 wpisz szukaną frazę/zakres cenowy/zakres ilościowy
 - 3.4.1 jeśli znaleziono idź do punktu 3.5
 - 3.4.2 jeśli nie znaleziono wróć do punktu 3.2
 - 3.5 przeglądaj pasujące produkty
- 4 Kup produkt
 - 4.1 wybierz tryb klienta
 - 4.1.1 wybierz wyszukiwarke,
 - 4.1.2 wybierz kryteria wyszukiwania
 - 4.1.3 wpisz szukaną frazę/zakres cenowy/zakres ilościowy
 - 4.1.3.1 jeśli znaleziono idź do punktu 4.1.4
 - 4.1.3.2 jeśli nie znaleziono wróć do punktu 4.1.1
 - 4.1.4 przeglądaj pasujące produkty
 - 4.1.5 kupuj wybrane produkty
 - 4.1.5.1 zaloguj się
 - 4.1.5.1.1 jeśli nie masz konta idź do punktu 4.1.5.2
 - 4.1.5.1.2 jeśli nieudane idź do 4.1.5.1
 - 4.1.5.1.3 jeśli udane idź do punktu 4.1.6
 - 4.1.5.2 zarejestruj się
 - 4.1.6 wpisz ilość zamawianych sztuk
 - 4.1.6.1 jeśli wystarczająca ilość sztuk na magazynie dodaj do koszyka
 - 4.1.6.2 jeśli brak wystarczającej ilości sztuk wróć do punktu 4.1.4

Scenariusze użycia dla sprzedawcy (administratora):

- 1 Wyszukaj produkty
 - 1.1 wybierz tryb sprzedawcy,
 - 1.2 zaloguj się,
 - 1.2.1 jeśli logowanie udane idź do punktu 1.3,
 - 1.2.2 jeśli logowanie nieudane wróć do punktu 1.1,
 - 1.3 wybierz wyszukiwarke,
 - 1.4 wybierz kryteria wyszukiwania,
 - 1.5 wpisz szukaną frazę/zakres cenowy/zakres ilościowy,
 - 1.5.1 jeśli znaleziono idź do punktu 1.6,
 - 1.5.2 jeśli nie znaleziono wróć do punktu 1.3,
 - 1.6 przeglądaj pasujące produkty.
- 2 Wyświetl listę zamówień
 - 2.1 wybierz tryb sprzedawcy,
 - 2.2 zaloguj się,
 - 2.2.1 jeśli logowanie udane idź do punktu 2.3,
 - 2.2.2 jeśli logowanie nieudane wróć do punktu 2.1,
 - 2.3 wybierz wyświetlanie listy zamówień,
 - 2.4 przeglądaj listę zamówień.
- 3 Wyświetl wszystkie produkty
 - 3.1 wybierz tryb sprzedawcy,
 - 3.2 zaloguj się,
 - 3.2.1 jeśli logowanie udane idź do punktu 3.3,

- 3.2.2 jeśli logowanie nieudane wróć do punktu 3.2,
- 3.3 wybierz wyświetlanie wszystkich produktów,
- 3.4 przeglądaj produkty.
- 4 Dodaj nowy produkt do bazy
 - 4.1 wybierz tryb sprzedawcy
 - 4.2 zaloguj się
 - 4.2.1 jeśli logowanie udane idź do punktu 4.3
 - 4.2.2 jeśli logowanie nieudane wróć do punktu 4.2
 - 4.3 wybierz dodawanie nowego produktu
 - 4.4 wypełnij formularz dodawania nowego produktu
 - 4.5 zdecyduj czy dodawać kolejne
 - 4.5.1 jeśli tak wróć do punktu 4.4
- 5 Edytuj istniejący produkt w bazie
 - 5.1 wybierz tryb sprzedawcy
 - 5.2 zaloguj się
 - 5.2.1 jeśli logowanie udane idź do punktu 5.3
 - 5.2.2 jeśli logowanie nieudane wróć do punktu 5.2
 - 5.3 wybierz wyszukiwarke
 - 5.4 wybierz kryteria wyszukiwania
 - 5.5 wpisz szukaną frazę/zakres cenowy/zakres ilościowy,
 - 5.5.1 jeśli znaleziono idź do punktu 5.4
 - 5.5.2 jeśli nie znaleziono wróć do punktu 5.6
 - 5.6 przeglądaj pasujące produkty
 - 5.7 wybierz edytuj przy wybranym produkcie
 - 5.8 wypełnij formularz edycji
- 6 Sortuj bazę produktów
 - 6.1 wybierz tryb sprzedawcy
 - 6.2 zaloguj się
 - 6.2.1 jeśli logowanie udane idź do punktu 6.3
 - 6.2.2 jeśli logowanie nieudane wróć do punktu 6.2
 - 6.3 wybierz sortowanie
 - 6.4 wybierz kryteria sortowania
 - 6.5 zdecyduj czy uaktualnić bazę

Opis przypadków użycia:

1. Wyświetl Koszyk (klient)
Służy do wyświetlania zamówionych przez danego klienta produktów. Za pomocą klawiszy 'z' (poprzedni) i 'x' (następny) można przeglądać produkty. Klawisz 'k' (kupuj) służy do kupowania produktu, klawisz 'q' (powrót) służy do powrotu do głównego menu klienta.
2. Wyświetl wszystkie produkty (klient i sprzedawca)
Funkcja służy do wyświetlania wszystkich produktów będących w ofercie sklepu internetowego. Za pomocą klawiszy 'z' (poprzedni) i 'x' (następny) można przeglądać produkty. Klawisz 'k' (kupuj) służy do kupowania produktu, klawisz 'q' (powrót) służy do powrotu do głównego menu klienta.
3. Wyszukaj produkty (klient i sprzedawca)
Otwiera menu wyszukiwania produktów w bazie. Mamy do wyboru pięć kryteriów wyszukiwania - według: nazwy, kategorii, opisu, ilości lub ceny. Wybierając nazwę, kategorię lub opis użytkownik powinien wprowadzić interesującą go frazę,

która ma zostać wyszukana odpowiednio w nazwie, kategorii lub opisie. W tym przypadku ciąg tekstowy sprawdzany jest znak po znaku dzięki czemu po wpisaniu w zapytaniu do wyszukiwarki frazy „wer” system znajdzie dla nas produkty „rowerek”, „wersalka”, „power”. Wybierając ilość lub cenę użytkownik powinien wpisać interesujący go zakres.

4. Kup produkt (klient)

Najważniejszy przypadek użycia sklepu internetowego - kupowanie. Użytkownik podczas wyświetlania produktu za pomocą klawisza 'k' (kupuj) składa zamówienie na dany produkt. Może zdecydować o ilości zamawianych sztuk. Produkt pojawia się w jego koszyku (liście zamówień), a ilość sztuk dostępnych w magazynie automatycznie pomniejszana jest o zamówioną przez klienta ilość sztuk.

5. Wyświetl listę zamówień (sprzedawca)

Służy do wyświetlania przez administratorów listy zamówień. Za pomocą klawiszy 'z' i 'x' możemy przesuwać się po zamówieniach od najstarszych do najnowszych. Administrator sklepu internetowego otrzymuje informacje o ilości zamówionych sztuk przez danego klienta.

6. Dodaj nowy produkt do bazy (sprzedawca)

Kolejny przypadek użycia wyłącznie dla sprzedawcy (administratora) sklepu. Za pomocą tej funkcji możemy wypełniając formularz dodać produkt do bazy. Po dodaniu każdego nowego produktu decydujemy czy dodawać dalej czy zakończyć dodawanie.

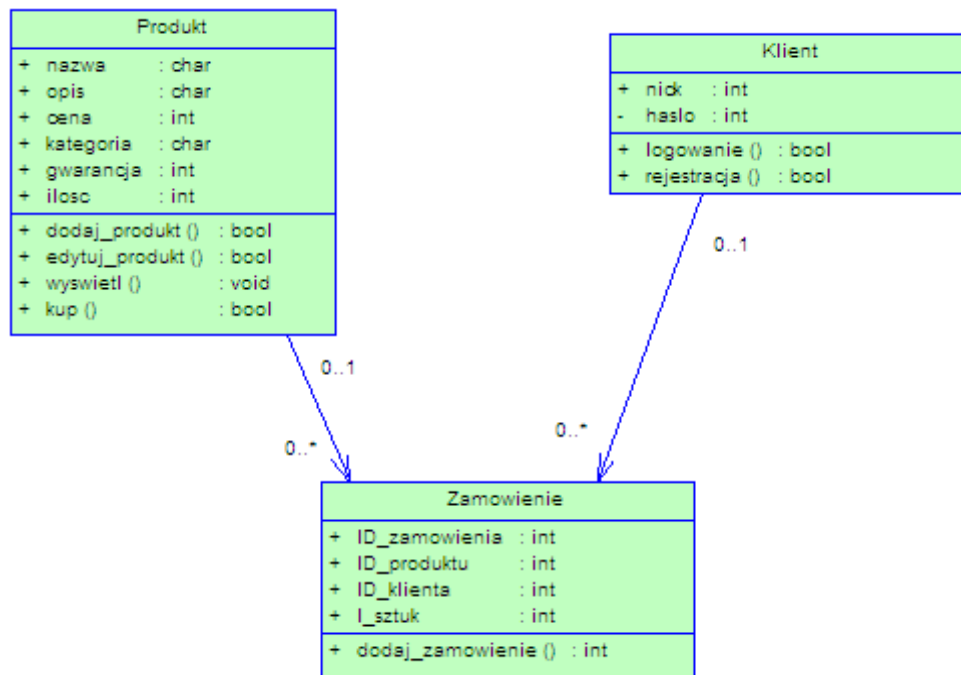
7. Edytuj istniejący produkt (sprzedawca)

Administrator podczas wyświetlania danego produktu za pomocą klawisza 'e' (edytuj) uruchamia funkcję edytuj odpowiedzialną za modyfikację istniejącego produktu w bazie produktów. Po dokonaniu zmian zostają wprowadzone zmiany do bazy.

8. Sortuj bazę produktów (sprzedawca)

Za pomocą tego przypadku użycia posortować produkty w bazie alfabetycznie według nazwy i kategorii oraz rosnąco według ilości oraz ceny. Po wykonaniu algorytmu sortowania administrator może zdecydować czy zapisać wyniki sortowania do bazy danych (pliku).

Model klas (class diagram)



Opis klas

1 Klasa Produkt

Obiekty klasy Produkt to poszczególne produkty dostępne w sklepie internetowym. Każdy produkt zawiera zmienne opisujące poszczególne cechy wystawionego do sprzedaży produktu:

- **nazwa** (tablica znaków char[60]) zawiera nazwę produktu,
- **opis** (tablica znaków char[300]) zawiera opis produktu,
- **cena** (liczba int) zawiera cenę,
- **kategoria** (tablica znaków char[60]) zawiera nazwę kategorii, do której należy produkt,
- **gwarancja** (liczba int) zawiera informację o długości gwarancji (m-ce),
- **ilosc** (liczba int) zawiera liczbę sztuk produktu w magazynie.

Oprócz cech klasa produkt zawiera jeden **konstruktor domyślny** oraz kilka **metod**:

- **dodaj_produkt()**; - konstruktor domyślny tworzący wypełniający tworzony obiekt danymi,
- **edytuj_produkta()**; - metoda aktualizująca zmienne w obiekcie,
- **wyswietl()**; - metoda wyświetlająca zawartość zmiennych danego obiektu,
- **kup()**; - metoda zmieniająca zmienną **ilosc** podczas procesu kupowania, ważny komponent funkcji globalnej **kupuj()**; metoda zwraca **true**, gdy można zakupić żadaną ilość produktu, bądź **false**, gdy nie można.

2 Klasa Klient

Obiekty klasy Klient to klienci zarejestrowani w systemie sklepu internetowego. Każdy obiekt zawiera w uproszczeniu zmienne **nick** i **haslo**. Zmienna **haslo** jest zahermetyzowana przed dostępem z zewnątrz klasy (enkapsulacja). Metoda **logowanie()**; sprawdza czy podane na wejściu **nick** i **haslo** pasują do bieżącego obiektu, zwraca **true** w przypadku natrafienia na zgodny obiekt, bądź **false** w przeciwnym przypadku. Konstruktor domyślny **rejestracja()**; wypełnia zmienne

nowotworzonego obiektu.

3 Klasa Zamowienie

Obiekty klasy **Zamowienie** to zamówienia złożone przez obiekty z klasy **Klient** na produkty z klasy **Produkt**. Każdy obiekt z klasy **Zamowienie** zawiera cztery zmienne:

- ID_zamowienia - unikalny identyfikator zamówienia ułatwiający pracę sprzedawcy sklepu,
- ID_produkta - zmienna asocjująca klasę **Zamowienie** z klasą **Produkt** - zawiera identyfikator produktu,
- ID_klient - zmienna asocjująca klasę **Zamowienie** z klasą **Klient** - zawiera identyfikator klienta.

Klasa **zamowienie** posiada jeden konstruktor domyślny:

- `dodaj_zamowienie()`; - wypełnia zmienne nowotworzonego obiektu.

Opis relacji

1. Relacja między klasami **Produkt** i **Zamowienie**.

Obiekt z klasy **Produkt** jest w relacji jeden do wielu z obiektem z klasy **Zamowienie**. Oznacza to, że obiekt klasy **Produkt** może być powiązany z wieloma obiektami klasy **Zamowienie**, natomiast obiekt klasy **Zamowienie** może mieć tylko jeden odpowiadający mu obiekt z klasy **Produkt**. Inaczej produkt może być zamówiony wiele razy, jednak pojedyncze zamówienie może opiewać tylko na jeden produkt.

2. Relacja między klasami **Klient** i **Zamowienie**.

Obiekt z klasy **Klient** jest w relacji jeden do wielu z obiektem z klasy **Zamowienie**. Oznacza to, że obiekt klasy **Klient** może być powiązany z wieloma obiektami klasy **Zamowienie**, natomiast obiekt klasy **Zamowienie** może mieć tylko jeden odpowiadający mu obiekt klasy **Klient**. Inaczej każdy klient może złożyć wiele zamówień, jednak każde zamówienie może opiewać tylko na jednego klienta.

3. Relacja między klasami **Produkt** i **Klient**.

Obiekty z tych klas nie są powiązane żadną relacją, ponieważ funkcję pośredniczącą spełniają obiekty klasy **Zamowienie**. Mówiąc inaczej obiekt **zamowienie** wiąże obiekty z klasy **Produkt** i **Klient**.

Wykorzystane mechanizmy paradygmatu obiektowego.

1. Enkapsulacja

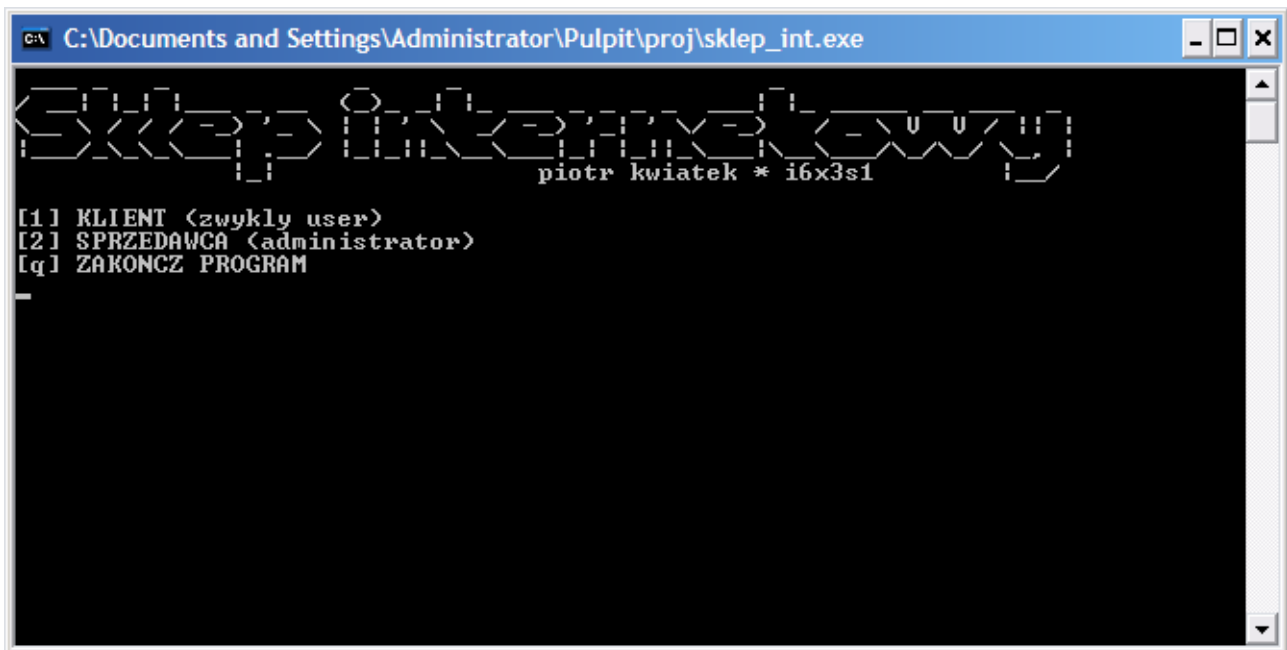
Inaczej hermetyzacja. Zapewnia, że obiekt lub funkcja zewnętrzna nie może zmieniać zmiennych innych obiektów w nieoczekiwany sposób. Tylko wewnętrzne metody obiektu są uprawnione do ich modyfikacji. Każdy typ obiektu prezentuje innym obiektom lub funkcjom swój „interfejs współpracy”.

Przykład:

Zmienna `haslo` w klasie `Klient` patrząc na jest sekcją `private` co powoduje jej niedostępność dla obiektów i funkcji zewnętrznych. Jedynie metoda `loguj()`, która jest upoważniona do odczytywania jej wartości i porównywania jej z zewnętrznymi zmiennymi.

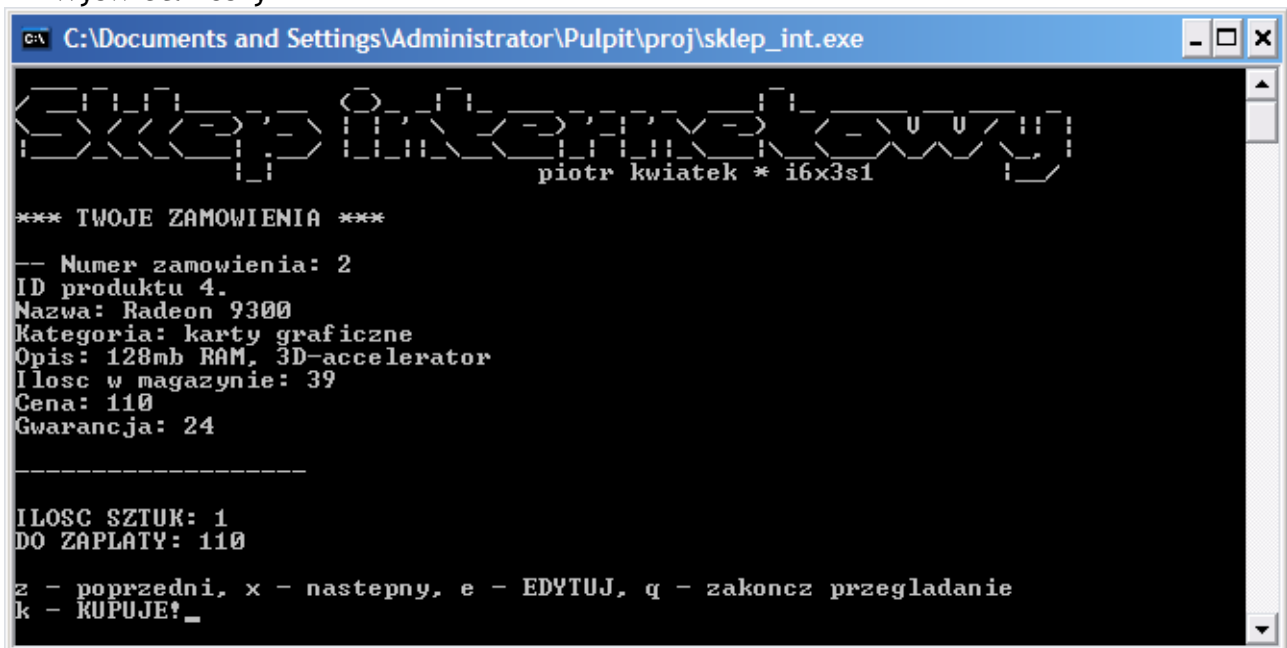
Zrzuty ekranów odpowiadające poszczególnym przypadkom użycia.

Ekran początkowy



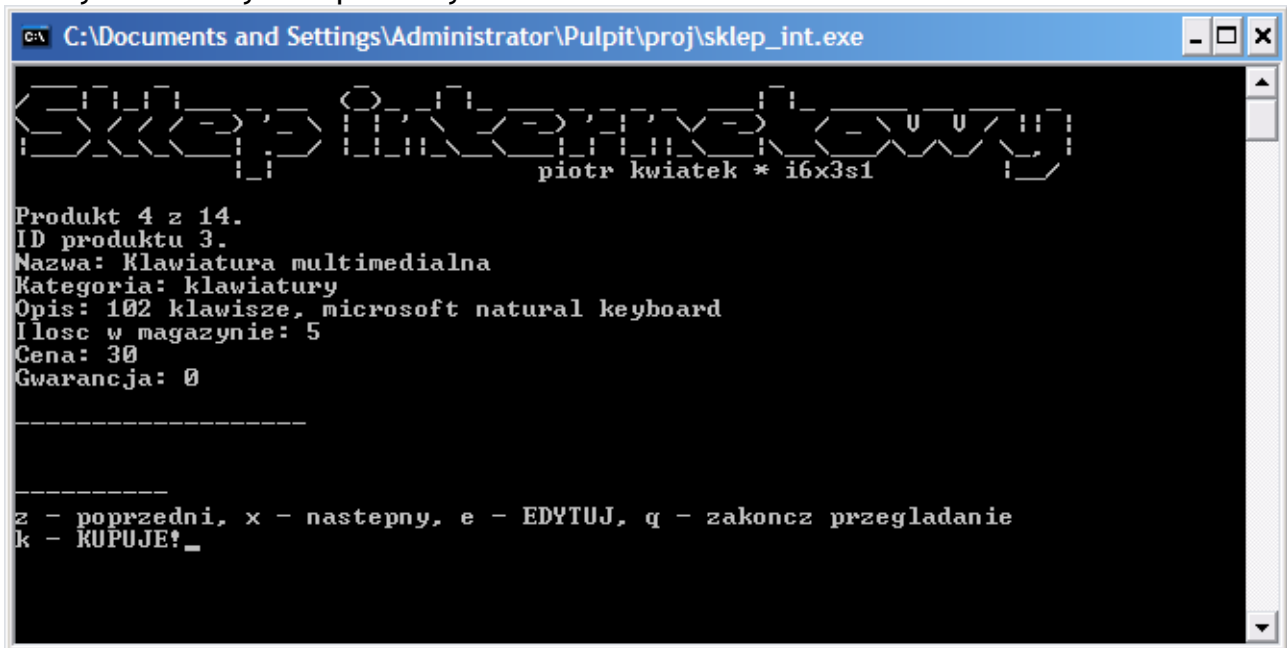
```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe
Sklep Internetowy
piotr kwiatek * i6x3s1
[1] KLIENT (zwykly user)
[2] SPRZEDAWCA (administrator)
[q] ZAKONCZ PROGRAM
```

- Wyświetl Koszyk

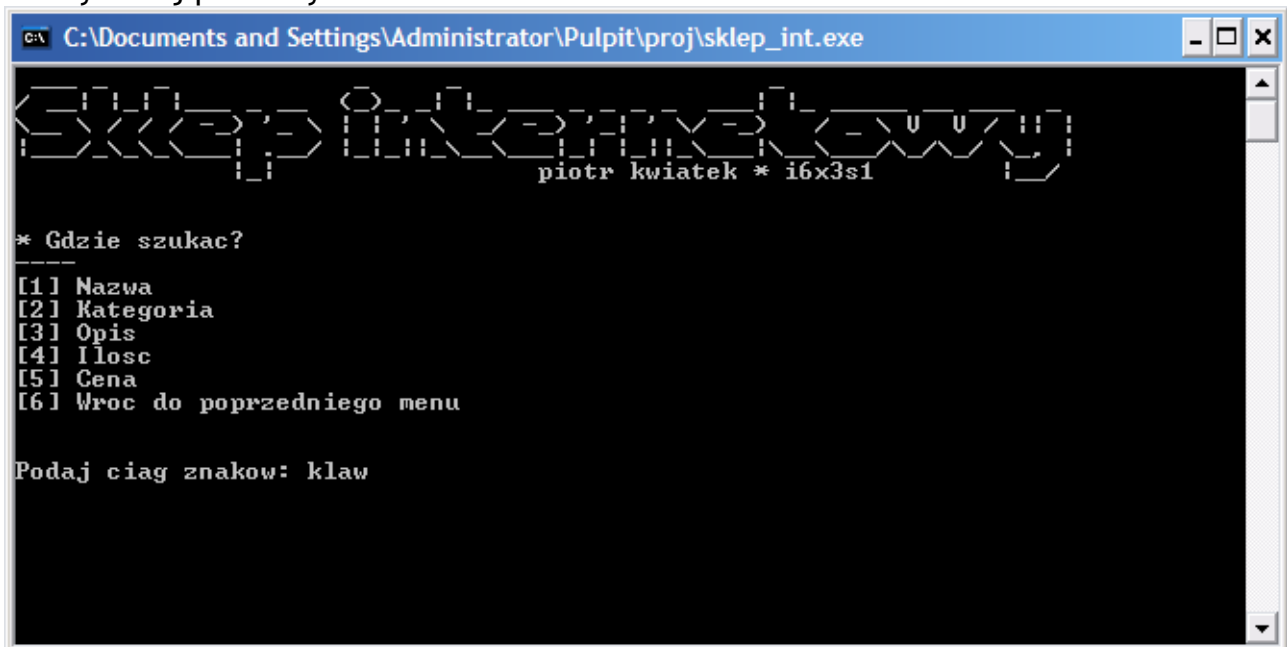


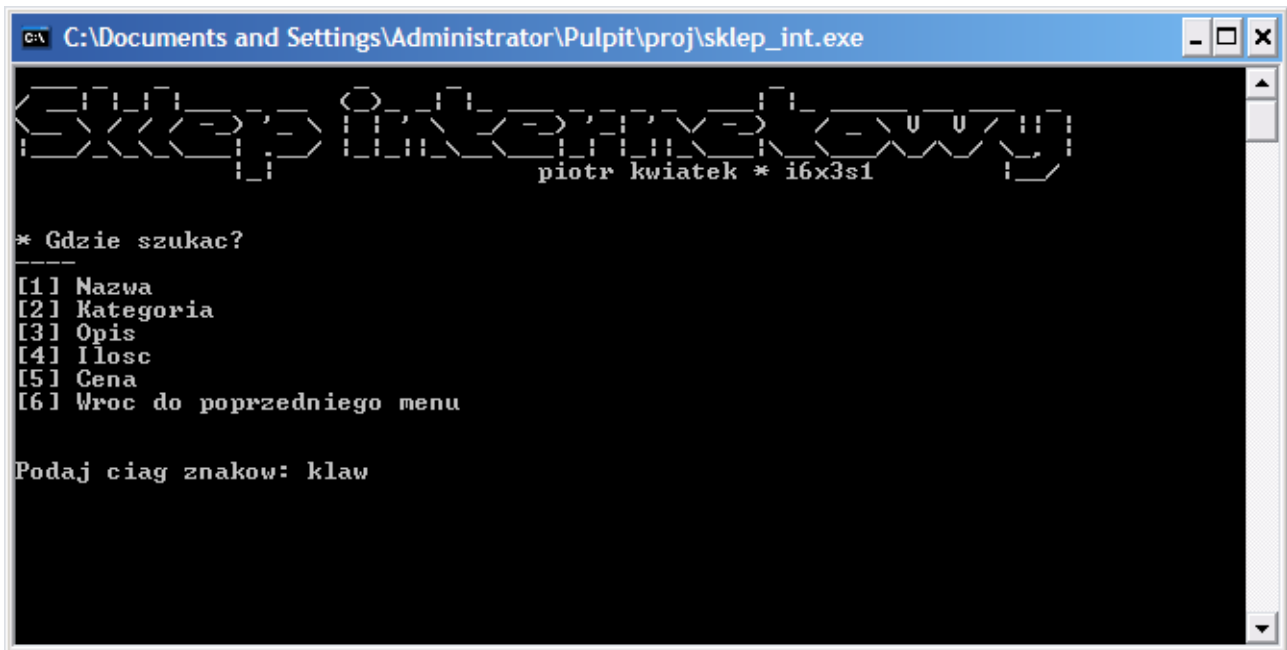
```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe
Sklep Internetowy
piotr kwiatek * i6x3s1
*** TWOJE ZAMOWIENIA ***
-- Numer zamówienia: 2
ID produktu 4.
Nazwa: Radeon 9300
Kategoria: karty graficzne
Opis: 128mb RAM, 3D-accelerator
Ilość w magazynie: 39
Cena: 110
Gwarancja: 24
-----
ILOSC SZTUK: 1
DO ZAPŁATY: 110
z - poprzedni, x - następny, e - EDYTUJ, q - zakoncz przegladanie
k - KUPUJE!_
```

- Wyświetl wszystkie produkty

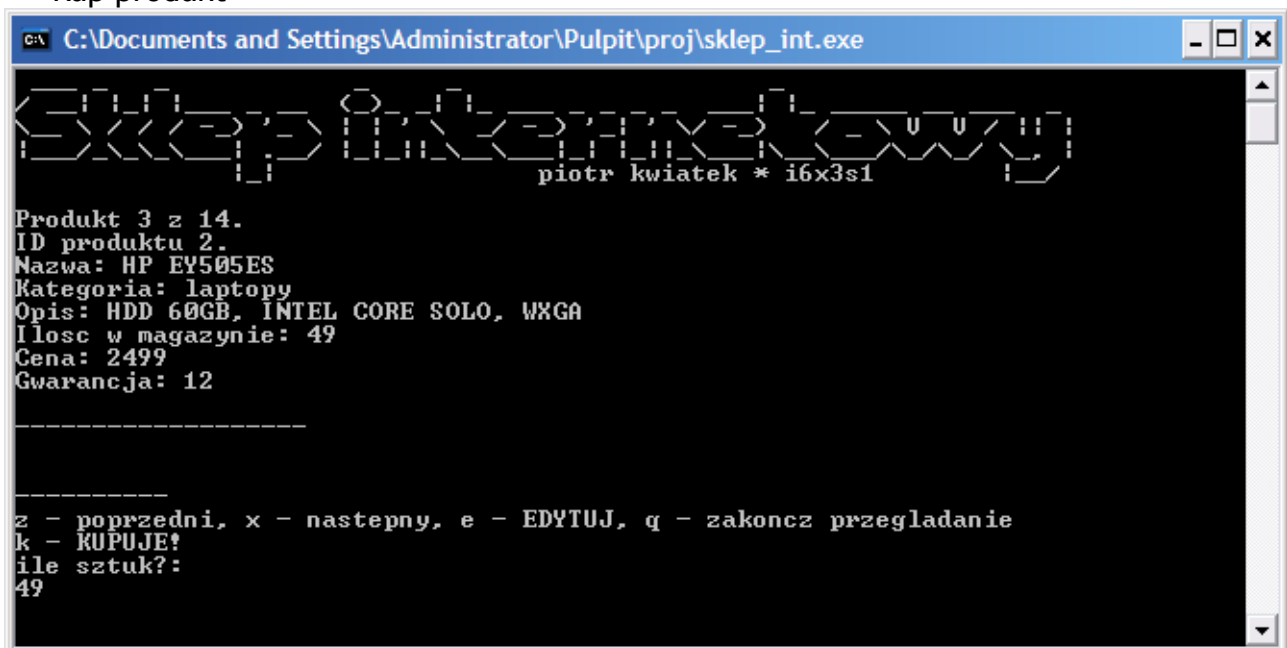


- Wyszukaj produkty

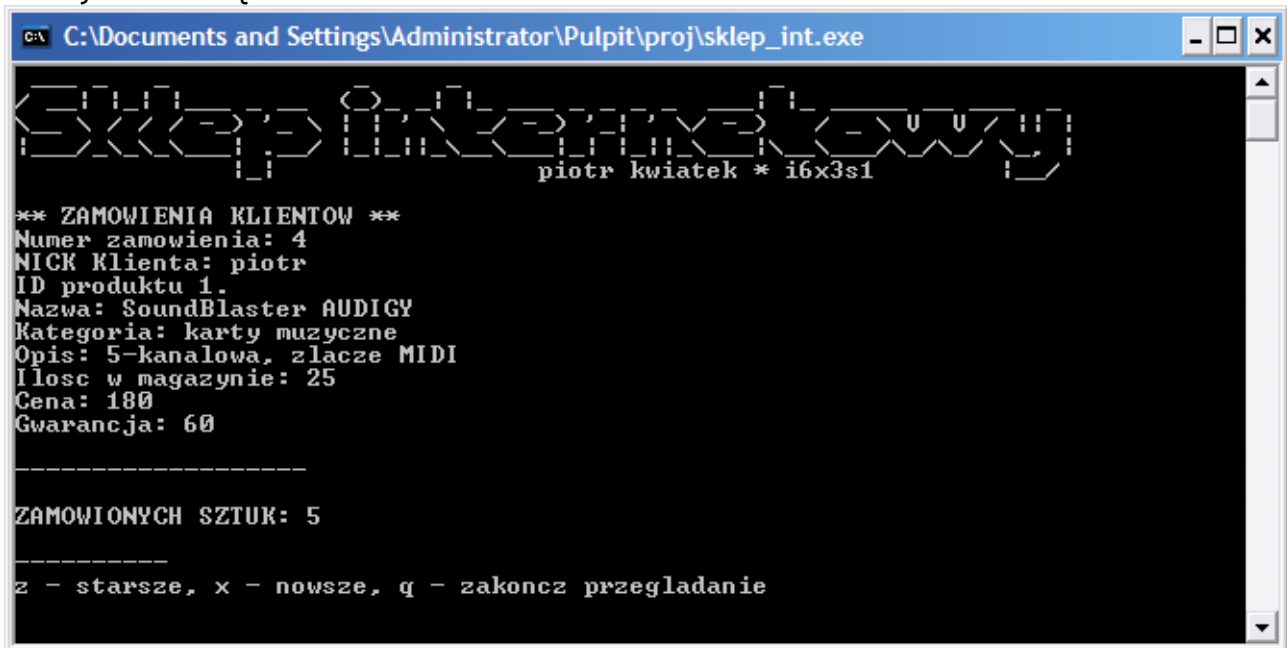




- Kup produkt



- Wyświetl listę zamówień



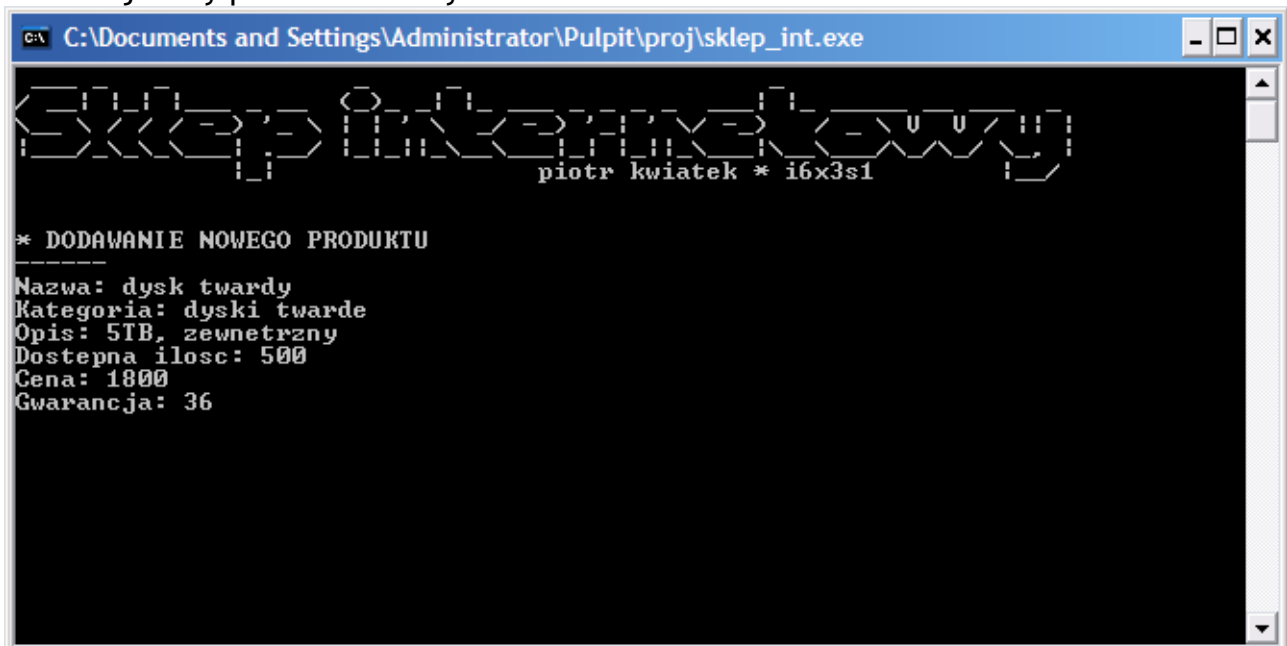
```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe

SKLEP ONLINE
piotr kwiatek * i6x3s1

** ZAMOWIENIA KLIENTOW **
Numer zamowienia: 4
NICK Klienta: piotr
ID produktu 1.
Nazwa: SoundBlaster AUDIGY
Kategoria: karty muzyczne
Opis: 5-kanalowa, zlacze MIDI
Ilosc w magazynie: 25
Cena: 180
Gwarancja: 60

-----
ZAMOWIONYCH SZTUK: 5
-----
z - starsze, x - nowsze, q - zakoncz przegladanie
```

- Dodaj nowy produkt do bazy



```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe

SKLEP ONLINE
piotr kwiatek * i6x3s1

* DODAWANIE NOWEGO PRODUKTU
-----
Nazwa: dysk twardy
Kategoria: dyski twarde
Opis: 5TB, zewnetrzny
Dostepna ilosc: 500
Cena: 1800
Gwarancja: 36
```

- Edytuj istniejący projekt

```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe  
-----  
ID produktu 3.  
Nazwa: Klawiatura multimedialna  
Kategoria: klawiatury  
Opis: 102 klawisze, microsoft natural keyboard  
Ilosc w magazynie: 5  
Cena: 30  
Gwarancja: 0  
-----  
Nazwa: klawiatura multimedialna  
Kategoria: rowery ;>_  
-----  
piotr kwiatek * i6x3s1
```

- Sortuj bazę produktów

```
C:\Documents and Settings\Administrator\Pulpit\proj\sklep_int.exe  
-----  
* Po czym sortowac?  
-----  
[1] Nazwa  
[2] Kategoria  
[3] Ilosc  
[4] Cena  
[5] Wroc do poprzedniego menu  
-----  
Posortowano. Zapisac do pliku? <t/n>_  
-----  
piotr kwiatek * i6x3s1
```

Propozycja rozbudowy programu

Klasy aplikacji można wzbogacić o szereg zmiennych i metod ułatwiających zakupy przedmiotów oraz pracę sprzedającym. Przydatne mogłyby być informacje mówiące o statusie realizacji zamówienia, dacie zamówienia, funkcje tworzące statystyki serwisu itp. Dodatkowo dobrym pomysłem byłoby zaimplementowanie dodatkowej klasy pracowników magazynu i spedycji, którzy także mieliby dostęp do systemu. Dzięki takiemu podziałowi obowiązków klient mógłby obserwować stan realizacji zamówienia oraz kontrolowanie operacji na każdym etapie. Obiekty klasy Produkt także mogłyby być wzbogacone o dodatkowe obiekty dziedziczące zawierające komentarze do produktów dodawane przez klientów. Dodatkowym udoskonaleniem byłoby przeniesienie bazy danych z plików do relacyjnej bazy danych przez co dostęp do danych mógłby następować bez niebezpieczeństwa utraty danych (hazard danych występujący podczas dostępu do tego samego pliku przez wiele uruchomionych aplikacji sklepu). Dzięki bazie danych znikłaby konieczność ładowania wszystkich danych do pamięci operacyjnej. Udoskonaleniami można by było wprowadzać wiele, jednak aplikacja napisana w języku C++ nigdy nie będzie tak funkcjonalna jak sklepy internetowe działające w Internecie oparte na języku PHP wykonywanym po stronie serwera wykorzystujące bazę np. bazę MySQL.