

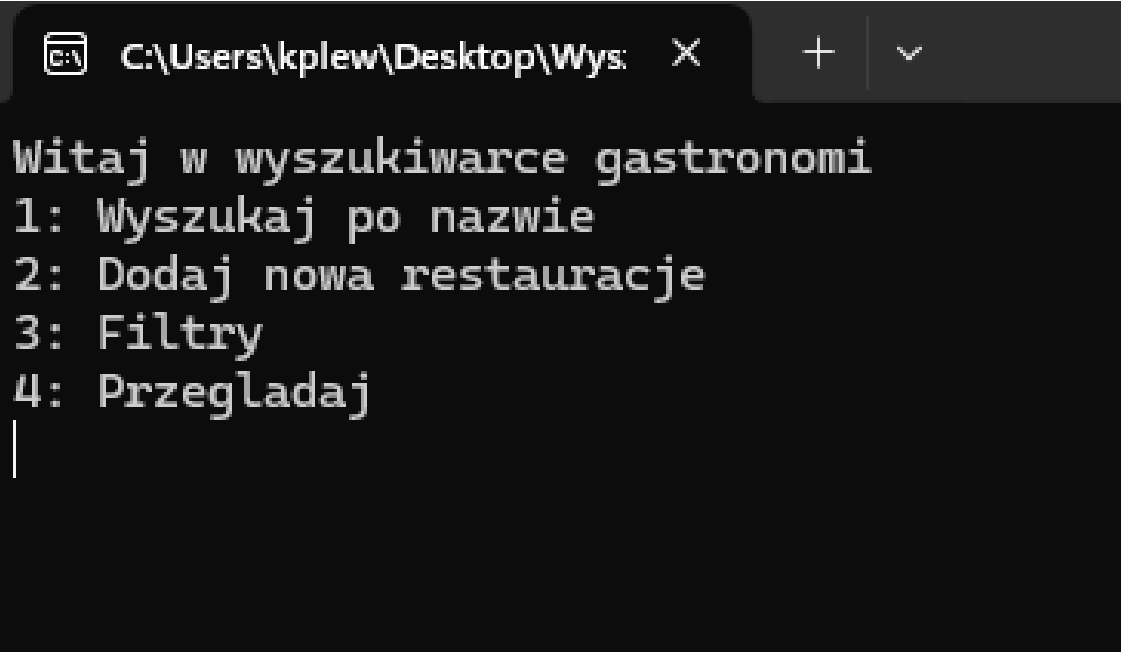
Sprawozdanie Jakub Plewiński Projekt PK2 Wyszukiwarka Gastronomi

Opis tematu:

Wyszukiwarka Gastronomi/ porównywarka gastronomi. Wyszukiwarka z różnymi opcjami takimi jak filtry, dodawanie restauracji , przegląd wszystkich, i wyszukiwanie po nazwie. Program oparty jest na programowaniu obiektowym co pozwala nam na łatwe zmienianie i dodawanie funkcjonalności. Temat jest również przyjemnym projektem na rozwój ogółu programowanie nie tylko aspektu obiektowego.

Interfejs:

Interfejs został stworzony w terminalu jako menu wielokrotnego wyboru z opcjami prowadzącymi nas w dalsza część projektu. menu zbudowane jest na podstawie klasy dziedziczącej z polimorfizmem. takie rozwiązanie jest bardzo wygodne ponieważ pozwala nam na dowolne dodawanie nowych odnóg interfejsu jedyne co trzeba zrobić to napisać klasę dziedziczącą po klasie Interfejs a potem wyświetlenie jej zawartości poprzez odpowiednie funkcje biblioteki iostream

A screenshot of a terminal window with a dark background. The title bar shows the file path 'C:\Users\kplew\Desktop\Wys:' and standard window controls. The terminal text is as follows:


```
Witaj w wyszukiwarce gastronomi
1: Wyszukaj po nazwie
2: Dodaj nowa restauracje
3: Filtry
4: Przeglądaj
|
```

po wybraniu opcji i zatwierdzeniu jej "Enterem" przejdziemy do następnego menu np.3 (filtry) następnie powtarzając wybór liczby i przycisk Enter jesteśmy w stanie przejść głębiej do interfejsu i używać zaimplementowanych funkcji jako wybór opcji. Dodawanie nowej funkcjonalności również nie powinno być problemem przez zastosowanie programowania obiektowego jesteśmy w stanie użyć wielu funkcji zaimplementowanych już wcześniej.



```
Witaj w wyszukiwarce gastronomi
1: Wyszukaj po nazwie
2: Dodaj nowa restauracje
3: Filtry
4: Przeglądaj
3
wybierz rodzaj filtru
1: cena w gore
2: cena w dol
3: ocena w gore
4: ocena w dol
5: rodzaj kuchni(kraj)
powrot: 0
|
```

Diagram klas

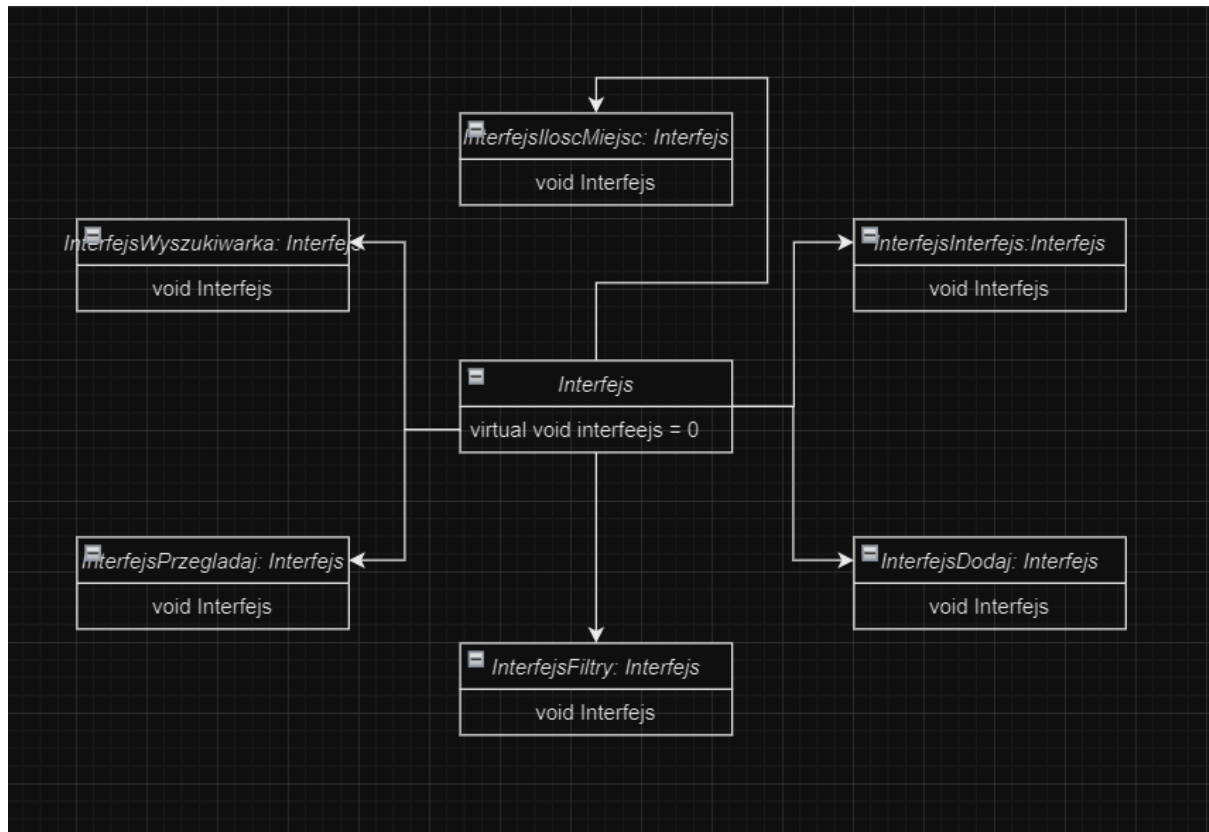
Klasa Restauracja trzymająca informacje o naszych restauracjach takie jak nazwa, cena, ocena, i rodzaj jako kraj z której pochodzi kuchnia . Metody w tej klasie są wykorzystywane do zmieniania i pobierania wartości z zmiennych aby mogły zostać użyte do filtrowania restauracji. Metoda Print odpowiada za wypisanie całego zbioru restauracji i wszystkich jego zmiennych

 Restauracja
nazwa
cena
ocena
rodzaj
get_nazwa
get_cena
get_ocena
get_rodzaj
set_nazwa
set_cena
set_ocena
set_rodzaj
print

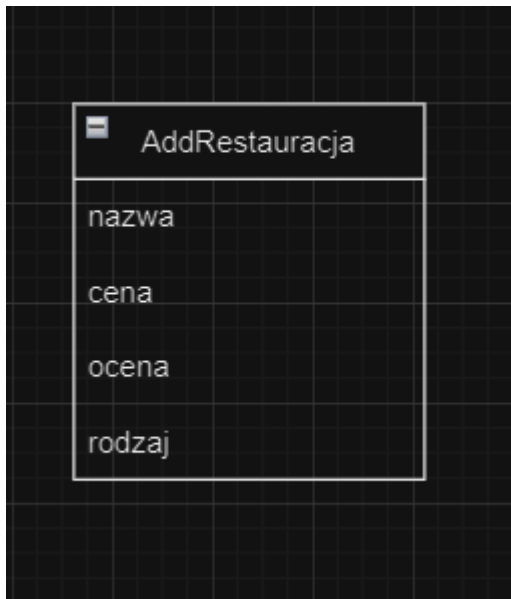
Klasy używane do filtrowania restaurac SortPrice sortuje cenę reprezentowaną przez znak \$
 metody dają nam możliwość sortowania cenę rosnąco i malejąco.
 SortRank pozwala nam na sortowanie rankingów podanego jako liczba zmiennoprzecinkowa również rosnąco lub malejąco

 SortPrice	 SortRank
PriceUp	RankUp
PriceDown	RankDown

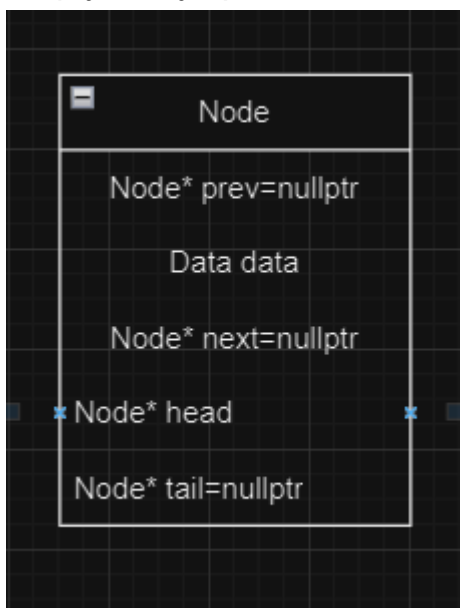
Klasa Dziedzicząca z polimorfizmem gdzie Interfejs i jego metoda są wirtualne te klasy obsługują Menu a ich metody pozwalają na wyświetlanie kolejnych części interfejsu wedle naszego życzenia. Przykładowo InterfejsWyszukiwarka dziedziczący po Interfejs ma metodę interfejs która pozwala nam na przejście do interfejsu wyszukiwania. Analogicznie reszta klas działa tak samo.



klasa pozwalająca na dodawanie nowych restauracji przez co później jesteśmy w stanie wpisać je do naszej bazy danych znajdującej się w pliku tekstowym. Klasa zawiera pola nazwa, cena, ocena, rodzaj . Wartości wpisywane są przez użytkownika według instrukcji wyświetlanej w konsoli



Następny diagram przedstawia budowę struktury danych jaką jest Doubly Linked List z polami takimi jak `next`, `prev` używanymi do zapisywania wskaźników do następnych nodów i `head`, `tail` które mówią nam który node jest pierwszy a który ostatni co pozwala nam na wyświetlanie struktury danych w optymalny sposób.



Struktury danych użyte w projekcie:

- Doubly Linked List

samodzielnie zaimplementowana struktura danych która przenosi dane po całym programie pozwalając wykonywać na tych danych akcje takie jak filtrowanie czy wyszukiwanie.

-stos

stos jest wykorzystywany podczas alokowania pamięci przy tworzeniu zwykłych zmiennych,
Przykładowo używane podczas tworzenia buforów potrzebnych dla pełnej funkcjonalności programu.

-zmienne

zmienne takiej jak bufor, zmienne do chwilowego przypisania wartości podczas sortowania i wiele więcej

Wnioski wyciągnięte z projektu:

Projekt nauczył mnie wiele na temat programowania obiektowego w języku C++, Programowanie obiektowe otwiera wiele drzwi na dalszy rozwój i pokazało bardzo efektywny sposób na implementację nowych funkcjonalności a potem przystępne zmienianie tej funkcjonalności bez nadpisywania wielu funkcji.

Podczas pisania projektu nie napotkałem na żadne większe problemy.