BAZY DANYCH 2 - DOKUMENTACJA PROJEKTU

Przetwarzanie własnych typów danych CLR UDT

Wykonał
Kamil Sudoł

1. Opis problemu.

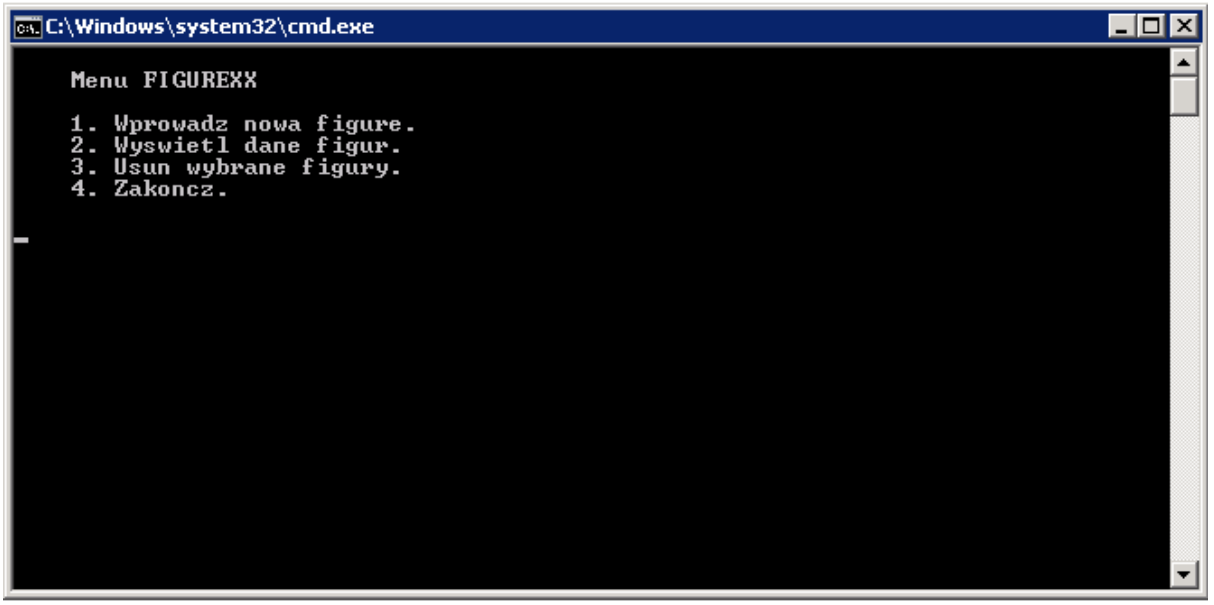
Celem projektu było opracowanie API oraz jego implementacji obsługującej wybrany zestaw typów własnych UDT w technologii CLR do obsługi typów złożonych. Opracowane API miało umożliwić wprowadzanie danych do zdefiniowanych struktur, wyszukiwanie danych w opracowanych strukturach oraz tworzenie odpowiednich raportów, jak również informować o błędnym ich wykorzystaniu.

Jako typy złożone, wybrane zostały dwuwymiarowe figury geometryczne, tj.:

- punkt,
- prosta,
- trójkąt,
- kwadrat,
- prostokąt,
- równoległobok,
- trapez,
- koło.

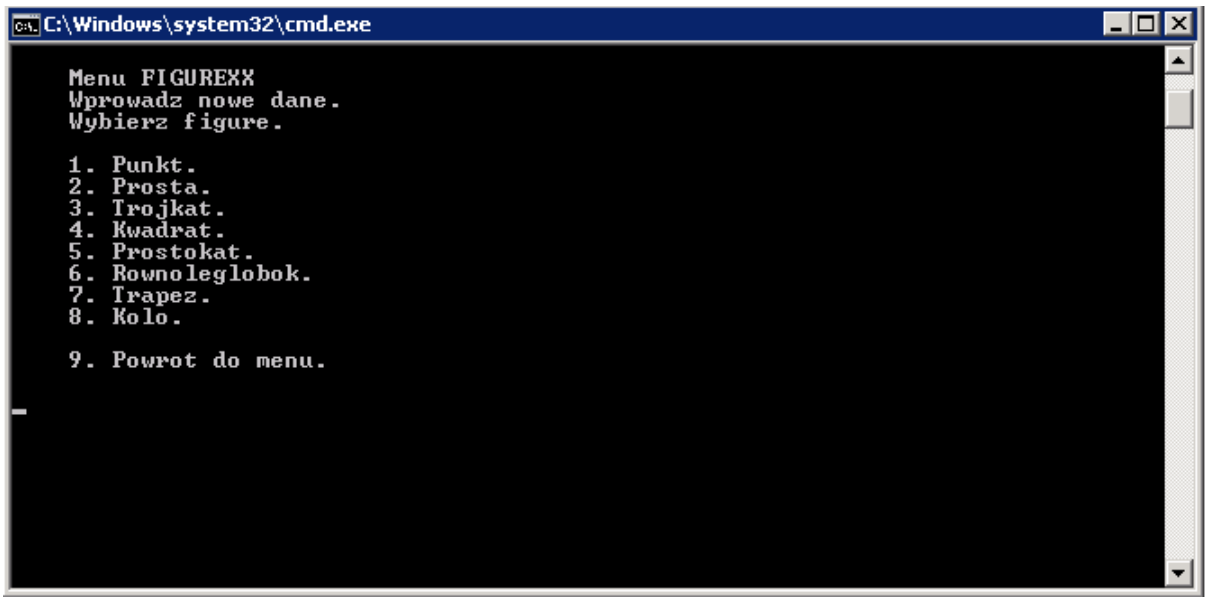
2. Opis funkcjonalności udostępnianej przez API.

W aplikacji konsolowej poruszamy się za pomocą wybierania udostępnionych opcji:



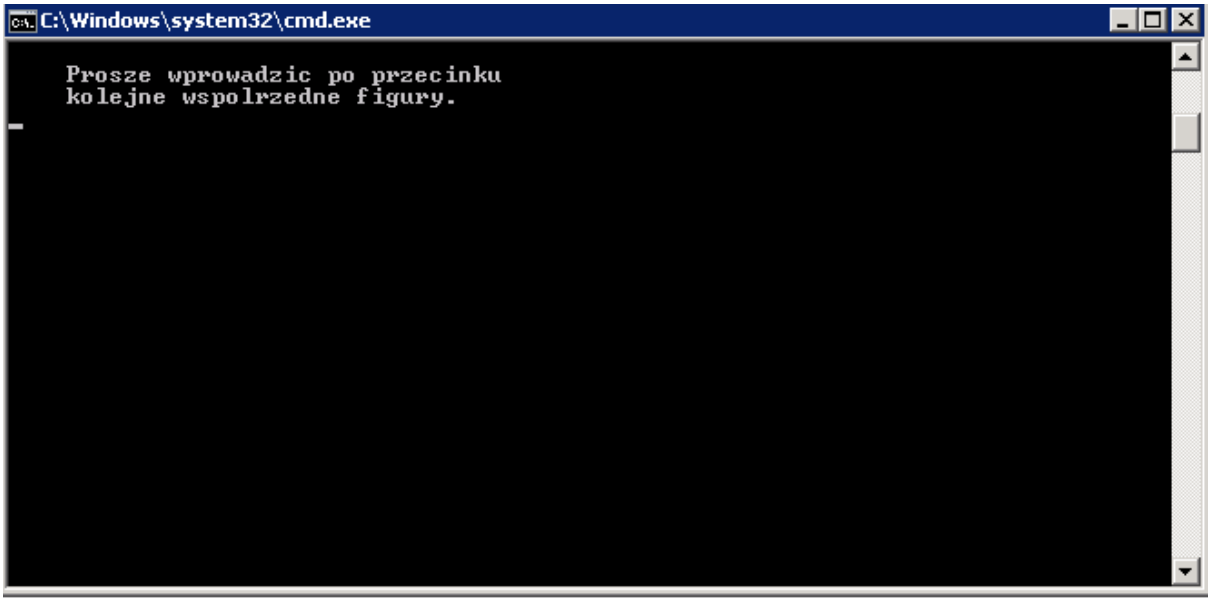
Obrazek 1: Menu główne aplikacji konsolowej.

Wybierając opcję “1” zostajemy przeniesieni do okna dialogowego wyboru figury, dla której mają zostać wprowadzone nowe dane.



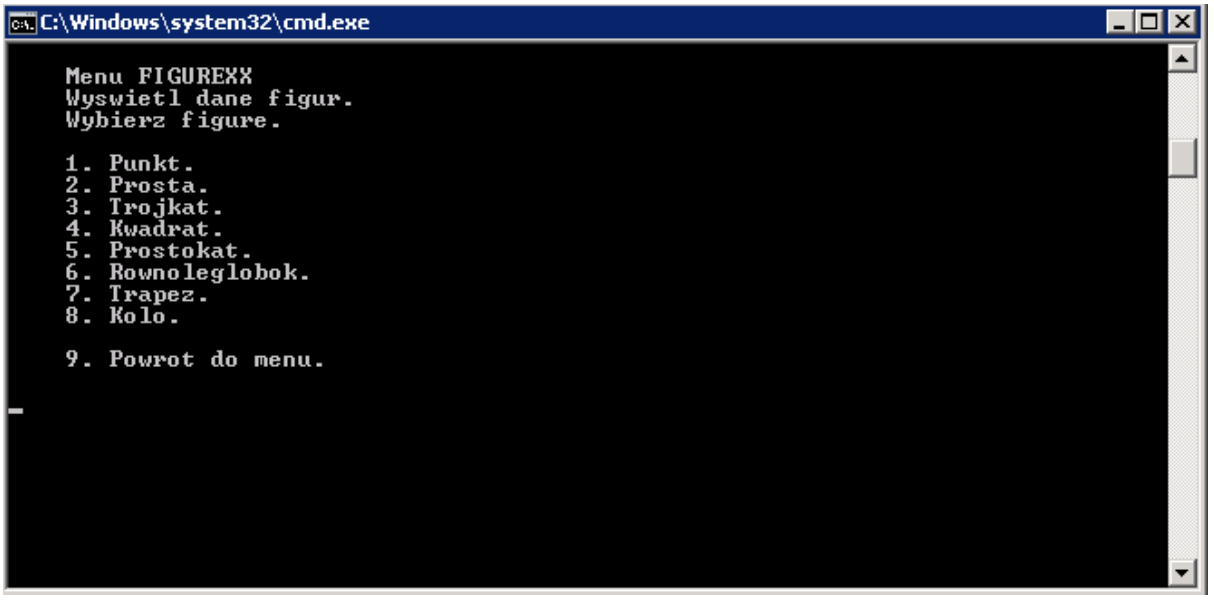
Obrazek 2: Menu wyboru figury, dla której mają zostać wprowadzone dane.

Po wyborze figury, aplikacja prosi nas, aby po przecinku wprowadzić kolejne współrzędne figury.



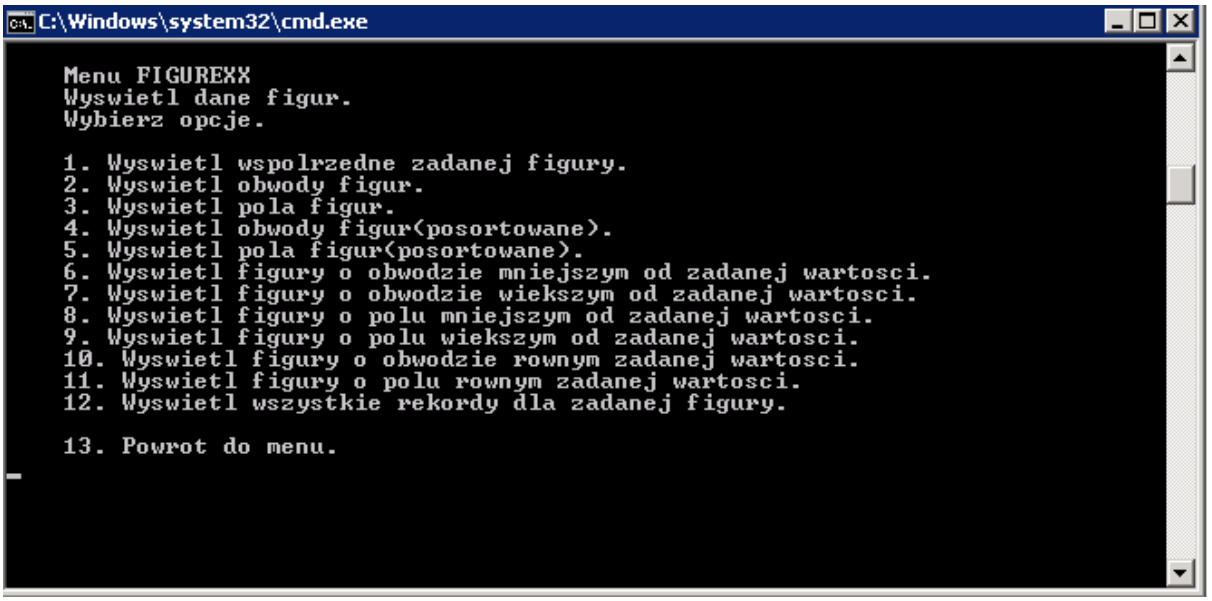
Obrazek 3: Informacja występująca przy prośbie podawania danych.

Wybierając opcję “2” w menu głównym ponownie zostajemy przeniesieni do okna dialogowego wyboru figury, dla której mają zostać wyświetlone dane.



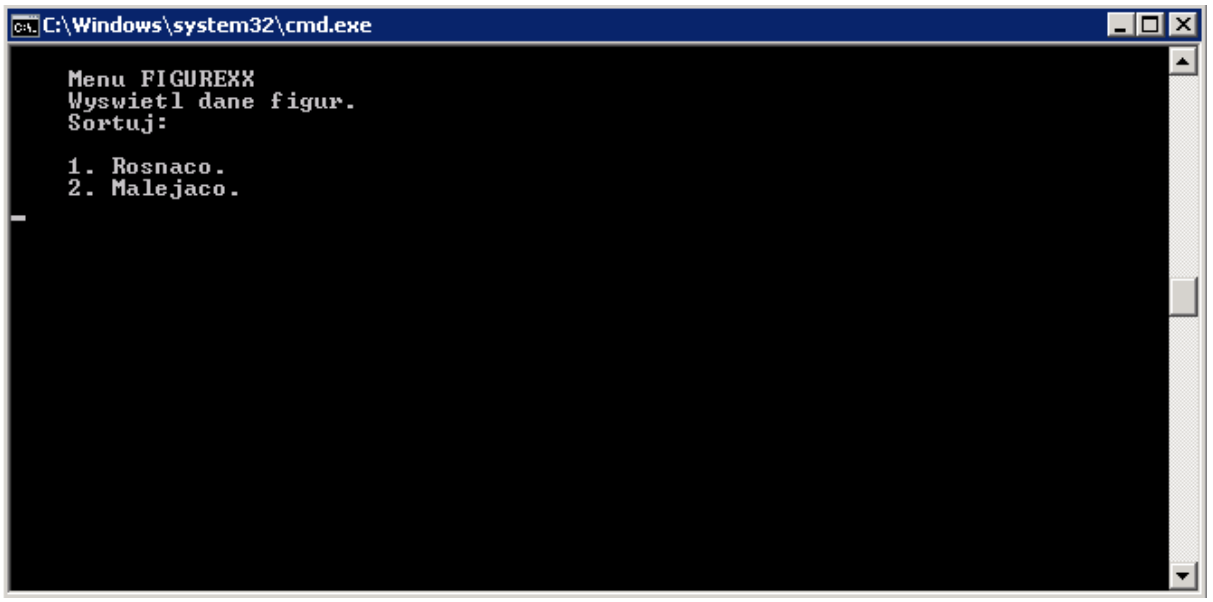
Obrazek 4: Menu wyboru figury, dla której mają zostać sporządzone raporty.

Po wyborze figury, aplikacja przenosi użytkownika do menu wyboru raportu.



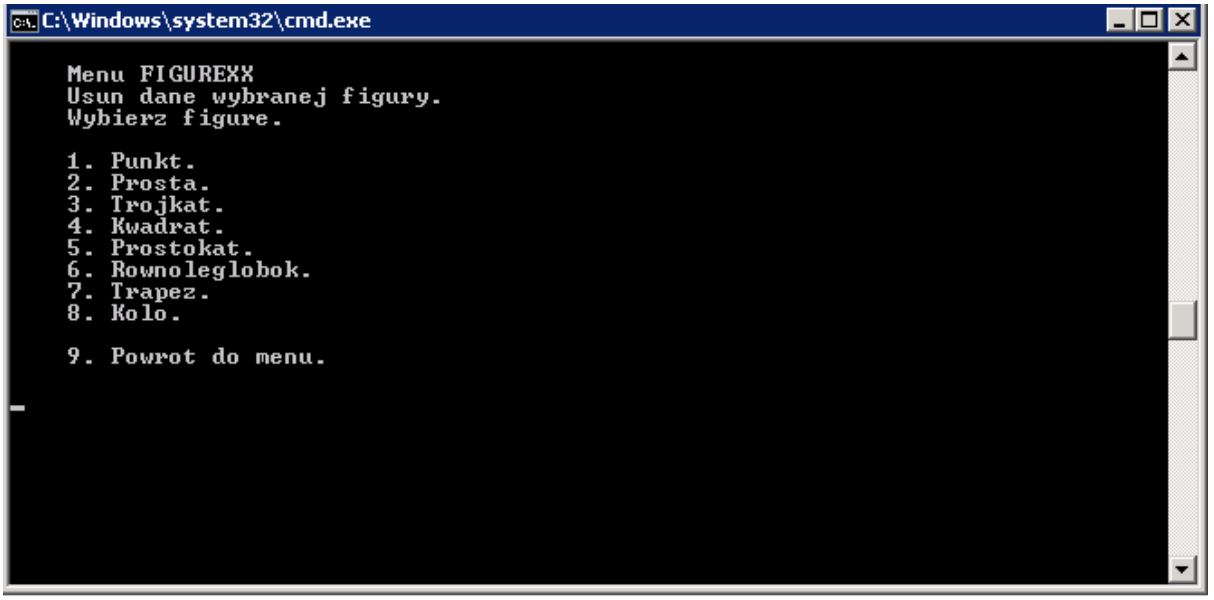
Obrazek 5: Menu wyboru raportu.

Wybierając opcje (1-3 oraz 12) aplikacja bez dalszych komunikatów wyświetli pożądane dane. W przypadku wyboru opcji (4-5) użytkownik zostanie dodatkowo poproszony o wybór sposobu sortowania (widoczne na obrazku nr 6), a w przypadku wyboru opcji (6-11) użytkownik zostanie poproszony o wprowadzenie wartości (widoczne na obrazku nr 9).



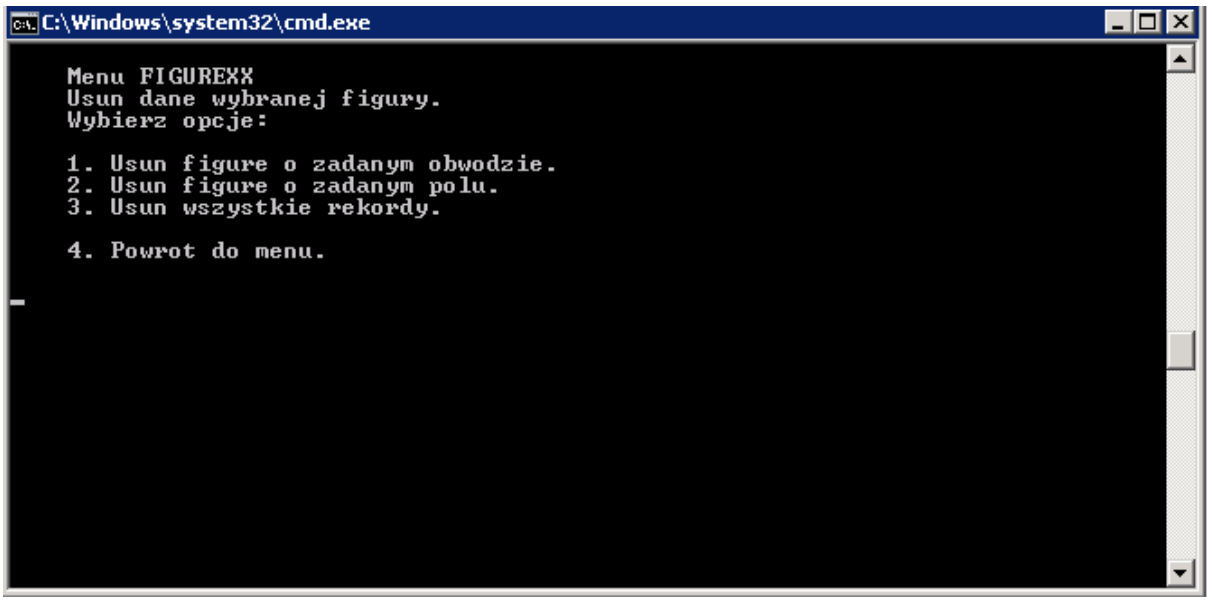
Obrazek 6: Prośba o wybór rodzaju sortowania.

Wybierając opcję “3” w menu głównym ponownie zostajemy przeniesieni do okna dialogowego wyboru figury, dla której mają zostać usunięte dane.



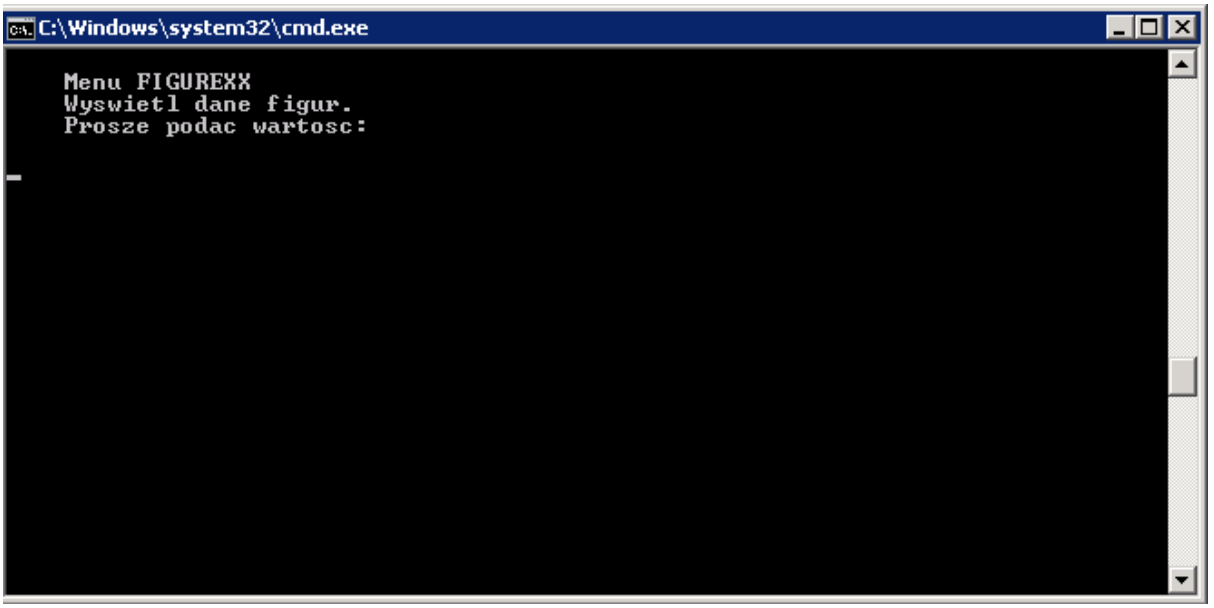
Obrazek 7: Menu wyboru figury, dla której mają zostać usunięte dane.

Po wyborze figury, aplikacja przenosi użytkownika do menu wyboru sposobu usunięcia danych.



Obrazek 8: Menu wyboru sposobu usunięcia danych figury.

Wybierając opcję “3” aplikacja bez dalszych komunikatów usunie wszystkie dostępne dane dla zadanej figury, natomiast przy wyborze opcji (1-2) aplikacja poprosi użytkownika o wprowadzenie żądanej wartości (widoczne na obrazku nr 9).



Obrazek 9: Prośba o wprowadzenie wartości przez interesanta.

3. Opis typów danych oraz metod udostępnianych w ramach API.

Obiekty UDT (*project_UDT*)

Wszystkie klasy figur zawierają pola przechowujące współrzędne punktów ich tworzących oraz ponadto w zależności od typu figury, przechowywane są długości boków, promienia itp.

Poza niektórymi wyjątkami, wszystkie klasy obiektów UDT zawierają ten sam zestaw metod, w które wchodzą:

static <typ figury> Parse(SqlString s) - metoda parsująca podany ciąg znaków do argumentów konstruktora figury.

string ToString() - metoda zwracająca opis figury, zawierający jej współrzędne.

string Obwod() - metoda zwracająca opis zawierający wartość obwodu figury.

double WyznaczObwod() - metoda zwracająca wartość obwodu figury.

string Pole() - metoda zwracająca opis zawierający wartość pola figury.

double WyznaczPole() - metoda zwracająca wartość pola figury.

void WyznaczBoki() (dla figur: trójkąt, kwadrat, prostokąt, równoległobok, trapez) - metoda wyznaczająca długości boków dla danej figury.

double WyznaczDlugosc() (dla figur: prosta, trójkąt, kwadrat, prostokąt, równoległobok, trapez) - metoda wyznaczająca długość odcinka pomiędzy dwiema zadanymi współrzędnymi.

void WyznaczPromien() (metoda istnieje tylko dla obiektu Kolo) - metoda wyznaczająca promień koła.

void Validator() (brak metody dla obiektu Punkt) - metoda sprawdzająca, czy z podanych współrzędnych można stworzyć zadaną figurę.

bool IsNull - defaultowa metoda sprawdzająca, czy zadany obiekt figury został utworzony.

Aplikacja konsolowa (*projectUDT_app*)

Na aplikację konsolową składają się następujące klasy:

DBConnection - klasa odpowiedzialna za połączenie z bazą danych, realizująca kwerendy itp.

Metody:

void SelectQuery(string query, List<string> attributes) - metoda realizująca polecenia SELECT.

void InsertQuery(string query) - metoda realizująca polecenia INSERT.

void DeleteQuery(string query) - metoda realizująca polecenia DELETE.

List<string> ExecuteQuery(string query, List<string> attributes) - bliźniacza metoda do metody *SelectQuery()*, jednak zwracającą listę danych. Metoda ta nie ma swojego zastosowania w aplikacji konsolowej, jest jednak wykorzystywana do przeprowadzania testów.

StopAppException - pomocnicza klasa dziedzicząca po klasie *Exception*, wykorzystywana, jako warunek zakończenia działania programu.

Figury - pomocnicza klasa zawierająca typ wyliczeniowy *Shape* przechowujący nazwy wszystkich obiektów UDT.

Program - klasa realizująca aplikację konsolową

Metody:

static void WyswietlFigury() - metoda wyświetlająca opcje wyboru dostępnych figur.

static string ResolveFigureType(int option) - metoda zwracająca nazwę figury.

static void WyswietlMenu() - metoda wyświetlająca menu główne.

static void ResolveMenuOption() - pomocnicza metoda ustalająca wybraną opcję z menu głównego.

static void WyswietlMenuInsert() - metoda wyświetlająca menu dodawania nowych figur do bazy.

static void WprowadzDane() - metoda pobierająca nowe dane figury od użytkownika oraz wysyłająca odpowiednie zapytanie do bazy danych.

static void WyswietlMenuSelect() - metoda wyświetlająca menu wyświetlania raportów figur z bazy.

static void WybierzPolecenie(string figura) - metoda wyświetlająca możliwe raporty do sporządzenia oraz wysyłająca odpowiednie zapytanie do bazy danych.

static string SortChoice(string thing) - pomocnicza metoda ustalająca sposób sortowania danych zwracająca odpowiednią klauzulę ‘ORDER BY’ do kwerendy.

static string WhereCondition(string figura, string thing, string delimiter) - metoda pobierająca od użytkownika pożądaną wartość, zwracającą odpowiednią klauzulę ‘WHERE’ do kwerendy.

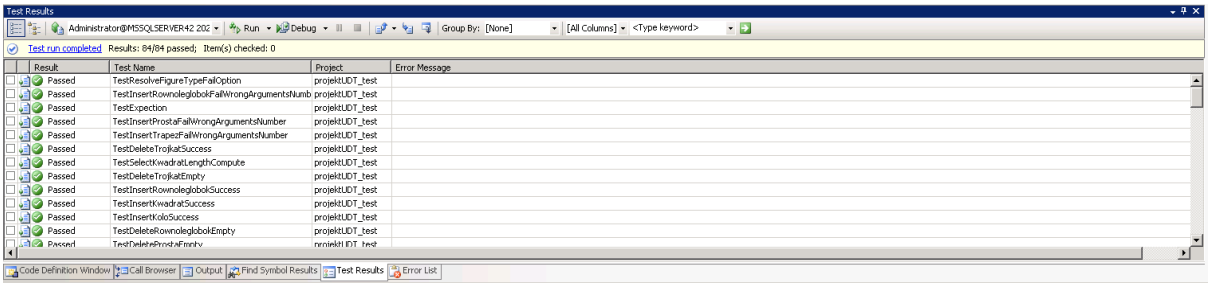
static void WyswietlMenuDelete() - metoda wyświetlająca menu usuwania danych figur z bazy.

static void DeleteOptions(string figura) - metoda wyświetlająca opcję usuwania danych figur.

static void Main(string[] args) - metoda startowa aplikacji konsolowej.

4. Prezentacja przeprowadzonych testów jednostkowych.

Łącznie przeprowadzono 84 testów, przetestowano działanie zarówno obiektów UDT, jak i elementów składających się na aplikację konsolową.



Obrazek 10: Pomyślne przejście wszystkich testów.

W celu uruchomienia testów wystarczy w projekcie ***projectUDT_app*** w zakładce Test wybrać Run -> All tests in solution.

5. Podsumowanie, uwagi.

Utworzenie bazy danych

Aby utworzyć bazę danych projektu, należy w programie SQL Server Managment Studio z folderu SQL uruchomić skrypt ***database_creation.sql***.

Deploy typów UDT

Aby wstawić do nowo utworzonej bazy danych typy UDT, należy w programie Visual Studio 2008 otworzyć projekt ***projectUDT***, a następnie wybrać zakładkę Build -> Deploy.

Utworzenie odpowiednich tabel do bazy danych

Aby utworzyć wymagane tabele w bazie danych, należy w programie SQL Server Managment Studio z folderu SQL uruchomić skrypt ***tables.sql***.

(**OPCJONALNIE**) W celu wypełnienia bazy przykładowymi danymi, należy w wyżej wymienionym programie z folderu SQL uruchomić skrypt ***insert_examples.sql***.

Aplikacja konsolowa

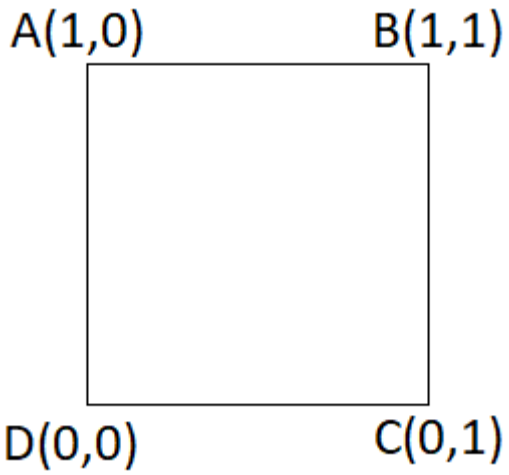
Aby uruchomić aplikację konsolową, należy w programie Visual Studio 2008 otworzyć projekt ***projectUDT_app***, a następnie wybrać zakładkę Debug -> Start without debugging.

Testy

Tak, jak było to wspomniane w podpunkcie 4 dokumentacji, w celu uruchomienia testów wystarczy w projekcie ***projectUDT_app*** w zakładce Test wybrać Run -> All tests in solution.

Uwagi

Aby wprowadzane współrzędne były zinterpretowane w poprawny sposób, należy je wprowadzać po kolei, np:



Obrazek 11: Przykładowe dane.

powyższy przypadek można wprowadzić do bazy na 4 różne sposoby: (A, B, C, D), (B, C, D, A), (C, D, A, B) oraz (D, A, B, C), najważniejsze jest tylko, aby zachować kolejność. Ponadto przy wprowadzaniu danych wszystkie współrzędne wprowadzamy po slash'u bez spacji, np. dla przypadku (A, B, C, D) będziemy mieli '1/0/1/1/0/1/0/0'. Części ułamkowe współrzędnych wpisujemy po przecinku, np. '1,2/0/1,1/1'.

Ponadto, wszelka walidacja danych wprowadzanych do bazy danych przeprowadzana jest na poziomie samej bazy danych, dlatego w przypadku, gdy użytkownik poda za mało/za dużo danych, niż

przewiduje to figura bądź dane będą innego typu (np. string), wówczas z bazy danych zostanie wyświetlony odpowiedni komunikat.

6. Literatura.

- stackoverflow.com
- https://newton.fis.agh.edu.pl/~antek/read_pdf.php?file=BD2_L09_CLR.pdf