INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA

PROGRAM NAUCZANIA INFORMATYKI Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Informatyka – poziom rozszerzony

Strukturalnie czy obiektowo – czyli droga do sukcesu

Andrzej Ptasznik

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Tytuł: ***Strukturalnie czy obiektowo – czyli droga do sukcesu***

Autor: ***Andrzej Ptasznik***

Redaktor merytoryczny: ***prof. dr hab. Maciej M. Sysło***

Materiał dydaktyczny opracowany w ramach projektu edukacyjnego
Informatyka – mój sposób na poznanie i opisanie świata.
Program nauczania informatyki z elementami przedmiotów matematyczno-przyrodniczych

www.info-plus.wwsi.edu.pl

infoplus@wwsi.edu.pl

Wydawca: Warszawska Wyższa Szkoła Informatyki
ul. Lewartowskiego 17, 00-169 Warszawa
www.wwsi.edu.pl
rektorat@wwsi.edu.pl

Projekt graficzny: *Marzena Kamasa*

Warszawa 2013

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

SCENARIUSZ TEMATYCZNY

STRUKTURALNIE CZY OBIEKTOWO – CZYLI DROGA DO SUKCESU

INFORMATYKA – POZIOM ROZSZERZONY

OPRACOWANY W RAMACH PROJEKTU:

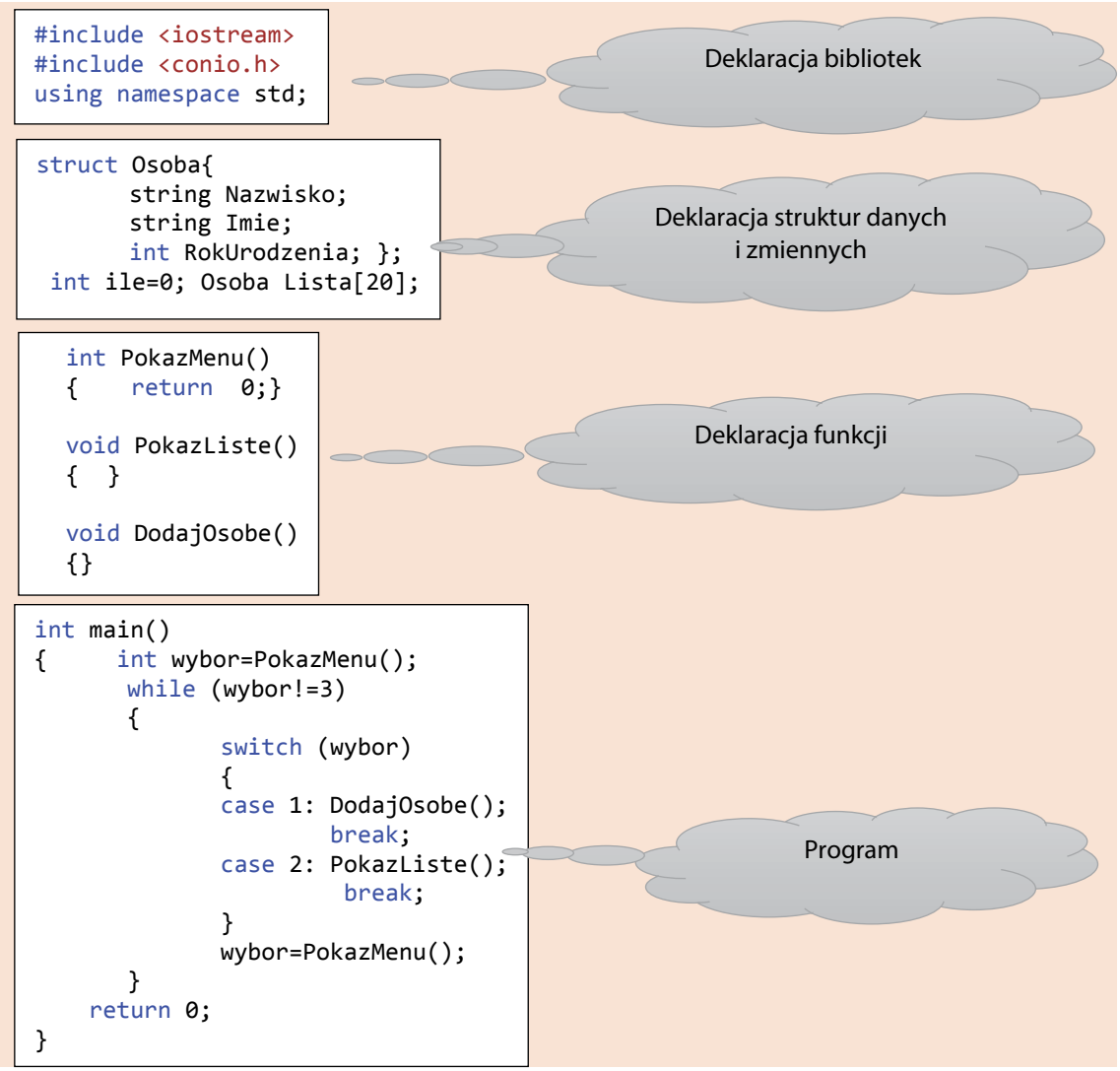
INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA.

PROGRAM NAUCZANIA INFORMATYKI

Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Streszczenie

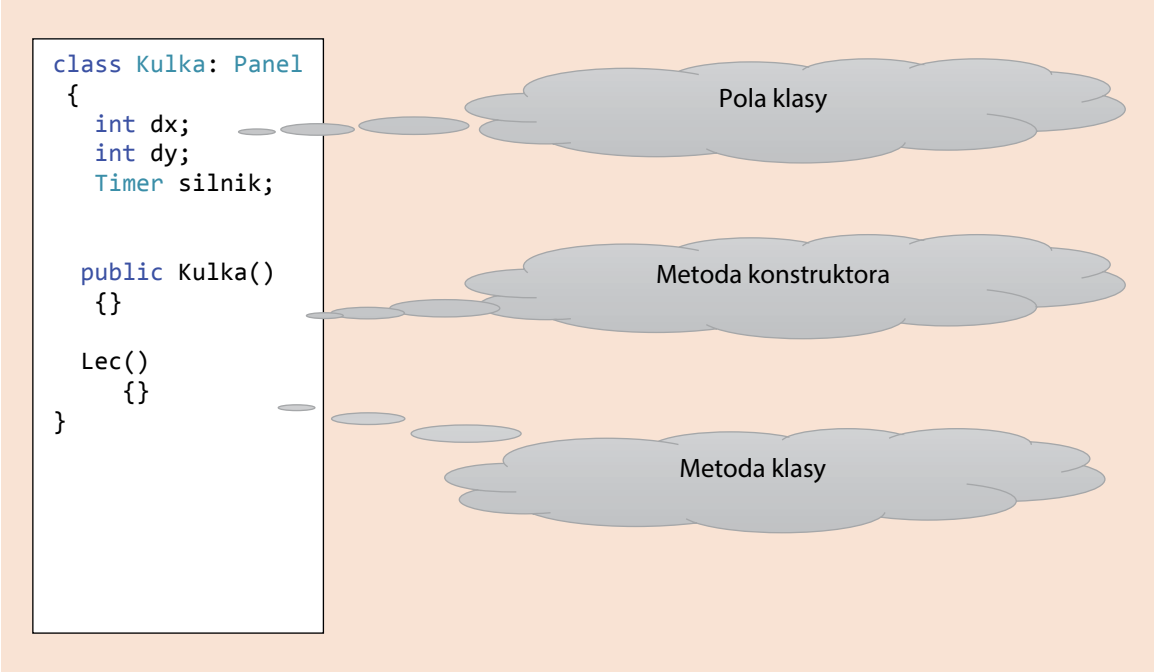
Programowanie obiektowe to nie tylko znajomość składni języka programowania - przede wszystkim to sposób myślenia o rozwiązywanym problemie. W ramach planowanych lekcji omówiony i zrealizowany zostanie przykładowy program, który dobrze wyjaśnia i pozwala zrozumieć jedną z podstawowych cech programowania obiektowego – hermetyzację. Na rysunku 1 pokazano fragment programu, w języku C++, w którym zaznaczono jego podział na kilka części.



Rysunek 1. Fragment programu.

W pokazanym na rysunku 1 przykładzie, struktury danych i funkcje są deklarowane oddzielnie bez jawnego powiązania z sobą, co jest uważane za jedną z głównych wad programowania strukturalnego. Program główny opisuje szczegółowo wszystkie ścieżki logiczne jego działania. W programowaniu obiektowym podstawową strukturą jest klasa, w której zawarte są struktury danych oraz funkcje, które z tych danych korzystają. Na podstawie definicji klasy w programie obiektowym tworzymy obiekty, które do realizacji przewidzianych zadań posiadają własne zasoby danych. Jest to przedstawiona w skrócie idea hermetyzacji, która jest podstawową cechą programowania obiektowego. Realizowany w ramach scenariusza przykład dobrze pokazuje istotę hermetyzacji i powinien ułatwić zrozumienie obiektowego podejścia do opisywania zadanych problemów. W ramach kolejnych lekcji scenariusza będziemy pisać program, w którym na ekranie będzie poruszała się dowolna liczba kolorowych kuleczek, przemieszczających się w różnych kierunkach i z różnymi prędkościami. Piłeczki będą odbijały się od krawędzi okna, w którym będą się poruszały, oraz będą odbijały się wzajemnie od siebie. Na bazie wykonanego wspólnie z uczniami programu, należy im zaproponować jego modyfikacje.

Na rysunku 2 pokazano przykładową definicję klasy.



Rysunek 2. Deklaracja klasy.

W ramach lekcji pierwszej należy omówić i wyjaśnić pojęcie klasy i obiektu, metody konstruktora, zdefiniować przykładową klasę i napisać program, który będzie tworzył obiekty zadeklarowanej klasy.

Czas realizacji

2 x 45 minut

Tematy lekcji

1 Klasa to podstawa
2 Symulacja ruchu, czyli szaleństwo na ekranie

LEKCJA NR 1

TEMAT: Klasa to podstawa

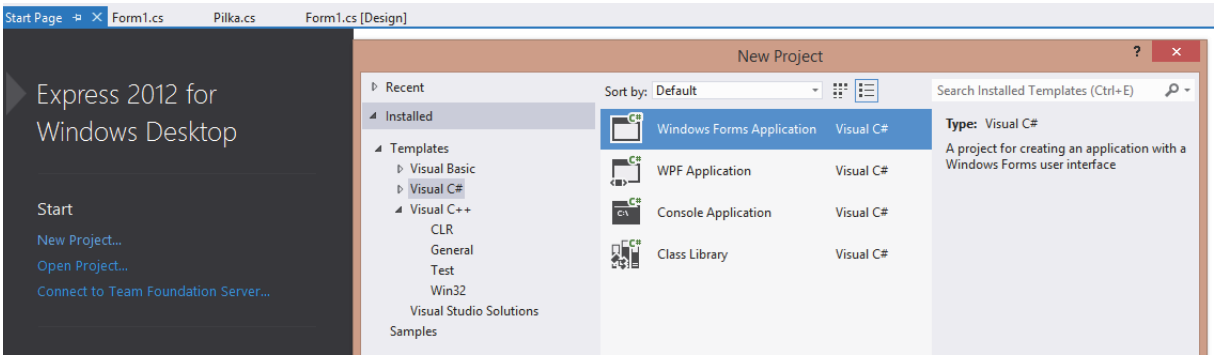
Streszczenie

Podstawowym zadaniem realizowanym w ramach lekcji jest zdefiniowanie klasy o nazwie Kulka, która wykorzystana będzie do symulacji chaotycznego ruchu kulek w zamkniętym obszarze. Kulki powinny odbijać się od krawędzi obszaru oraz odbijać się od innych kulek (symulacja zderzenia).

Zadanie realizujemy w następujących krokach:

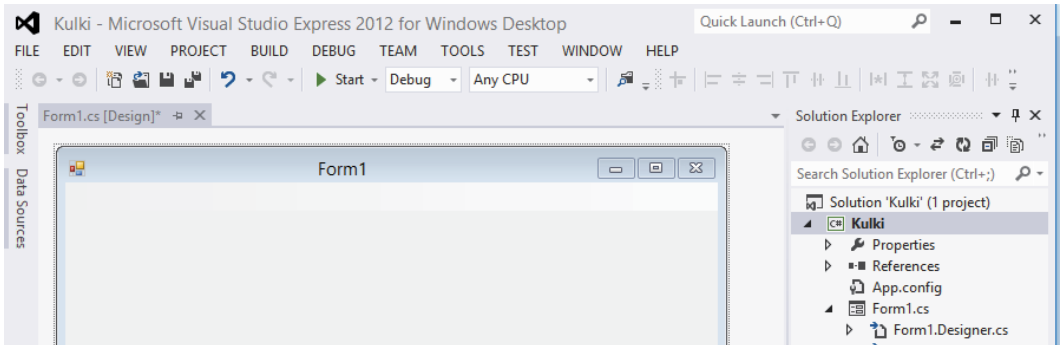
1. Uruchomienie środowiska Visual Studio i wybór odpowiedniego wzorca projektu.

Na rysunku 2 pokazano wybór odpowiedniego wzorca, po uruchomieniu programu Visual Studio 2012.



Rysunek 2. Wybór Windows Forms Application w środowisku Visual Studio 2012.

Wybieramy język programowania C#, który jest obiektowym językiem programowania opracowanym przez firmę Microsoft oraz wzorzec Windows Forms Application. Po uruchomieniu opcji wskazanej na rysunku 2 zostanie wyświetlony podstawowy widok projektu (rysunek 3). Podstawowy widok projektu zawiera formularz, który po uruchomieniu programu będzie jego oknem, w którym prezentowane będzie działanie. Widoczne jest także okno o nazwie Solution Explorer, w którym jest dostęp do wszystkich elementów składowych projektu oraz okno Properties, które zapewnia dostęp do właściwości i metod wybranego obiektu. Po uruchomieniu, w oknie Properties widoczne są właściwości i metody formularza.

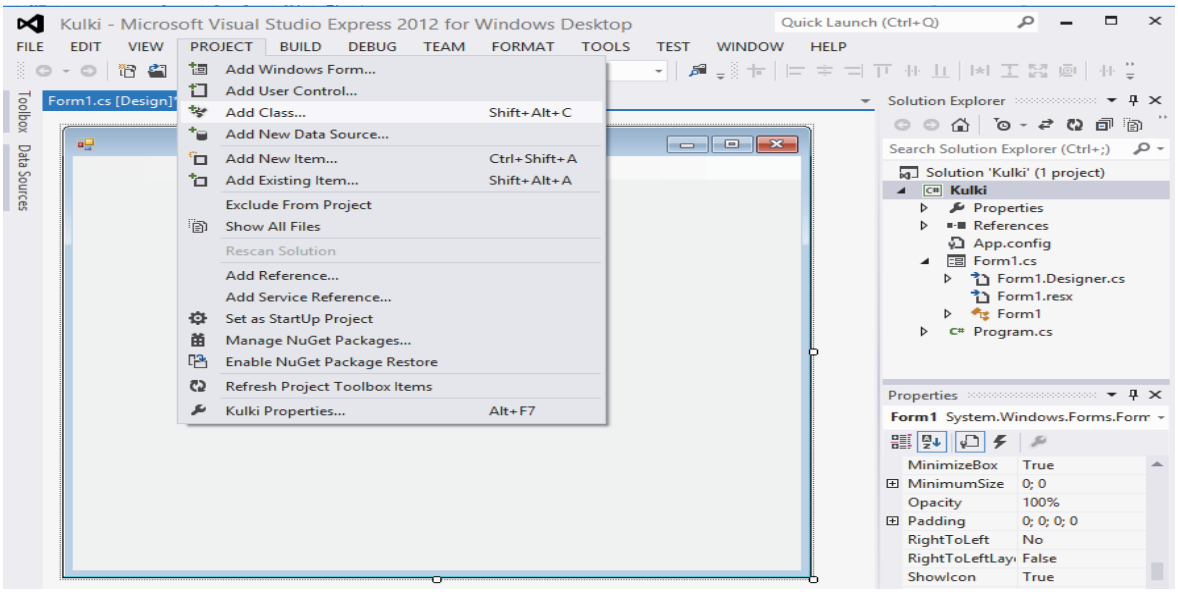


Rysunek 3. Widok okien projektu.

Możemy teraz przystąpić do definiowania klasy Kulka.

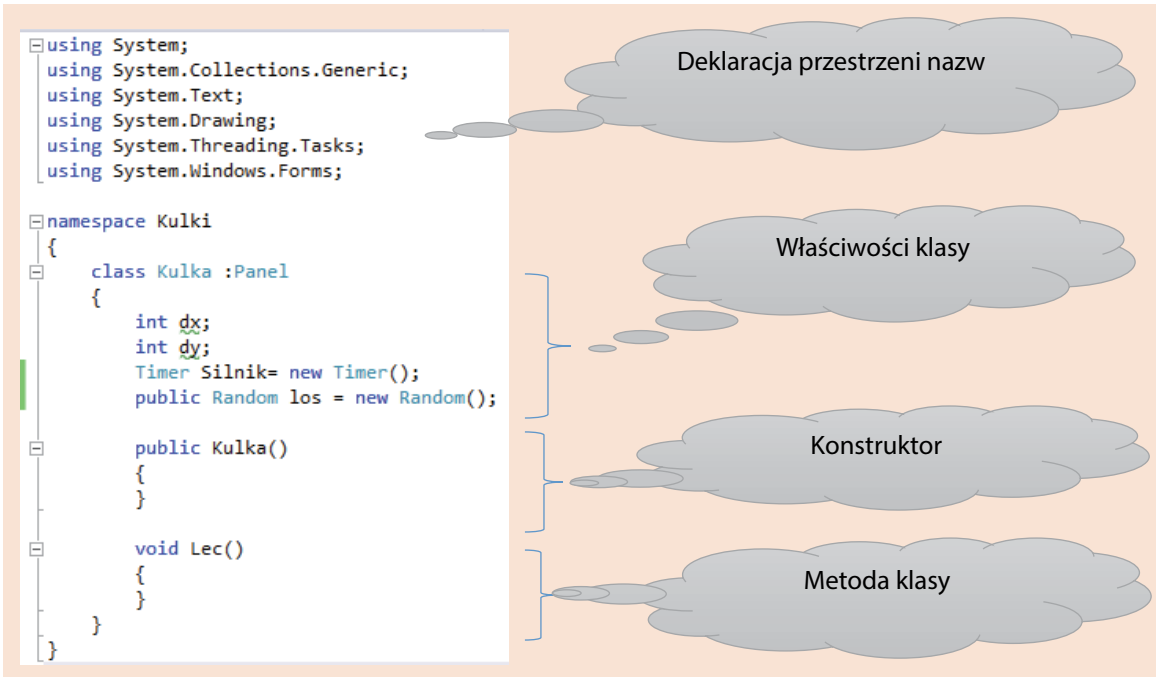
2. Otworzenie pliku i zdefiniowanie klasy o nazwie Kulka.

W kolejnym kroku wybieramy menu Project i wybieramy opcję Add Class, jak pokazano na rysunku 4.



Rysunek 4. Wybór zadania Add Class.

Po wyborze opcji otrzymujemy okno, w którym będziemy mogli definiować klasę Kulka. Na rysunku 5 pokazano kod deklaracji klasy Kulka. W klasie zdefiniowano cztery pola oraz dwie metody.



Rysunek 5. Deklaracja klasy Kulka.

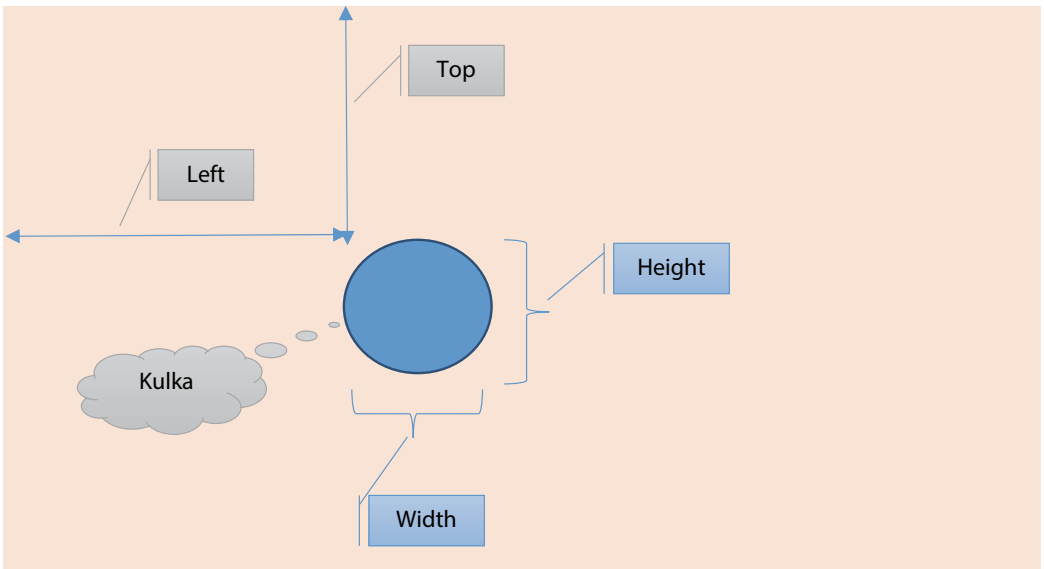
Pola dx i dy przechowują wartości wektora przesunięcia kulki i będą wykorzystywane przy implementacji metody Lec, która będzie opisywała ruch kulki na ekranie. Pole Silnik klasy Timer będzie wykorzystane do

symulacji ruchu, a pole los klasy Random będzie umożliwiało generowanie wartości losowej. Metoda Kulka, której nazwa jest taka sama jak nazwa klasy, to konstruktor, czyli metoda odpowiedzialna za tworzenie obiektu danej klasy. W dalszych krokach, w konstruktorze ustalony zostaje kolor tworzonego obiektu jako losowa kombinacja trzech składowych RGB. Kolejnymi elementami, których wartość ustalamy w konstruktorze to wektor przesunięcia dx i dy oraz rozmiar kulki reprezentowany przez właściwości Width i Height. Na rysunku 6 pokazano kod klasy Kulka z gotowym kodem konstruktora.

```
namespace Kulki
{
    class Kulka :Panel
    {
        int dx;
        int dy;
        Timer Silnik= new Timer();
        public Random los = new Random();

        public Kulka()
        {
            BackColor=Color.FromArgb(los.Next(255), los.Next(255), los.Next(255));
            dx = 5 - los.Next(10);
            dy = 5 - los.Next(10);
            Width = 25;
            Height = 25;
        }

        void Lec()
        {
        }
    }
}
```

Rysunek 6. Deklaracja klasy Kulka z kodem konstruktora.

Przedstawiona na rysunku 6 deklaracja klasy Kulka nie jest pełnym rozwiązaniem zadania, ponieważ nie zaimplementowaliśmy metody Lec, która będzie realizowała symulację ruchu kulki i odbicia od ścian okna oraz od innych kulek. Na rysunku 7 pokazano interpretację właściwości Top, Left, Height i Width, które określają rozmiar kulki i jej położenie względem okna, w którym będą wyświetlane.

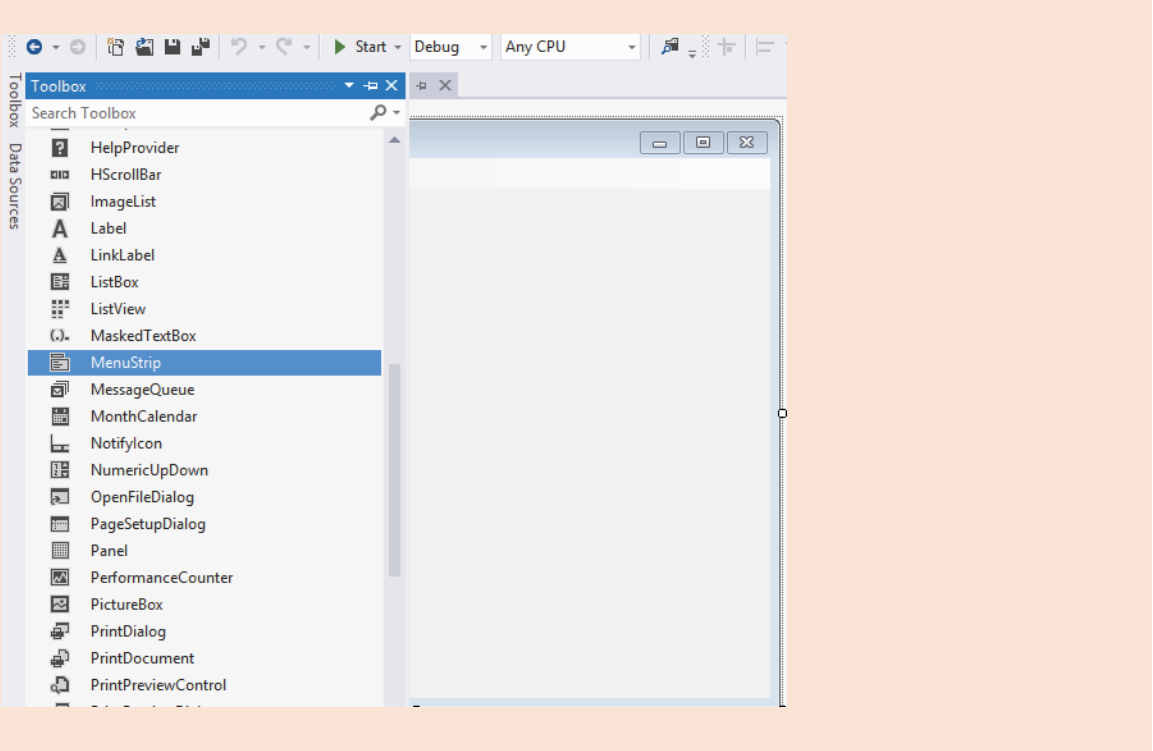


Rysunek 7. Interpretacja wybranych właściwości klasy Kulka.

W tym miejscu należy wyjaśnić skąd klasa Kulka posiada właściwości Top, Left, Height i Width, mimo że w deklaracji klasy nie widzimy ich jawnej deklaracji. Klasa Kulka została zadeklarowana jako klasa dziedzicząca z klasy o nazwie Panel, a to znaczy, że omawiane właściwości zostały odziedziczone z klasy Panel.

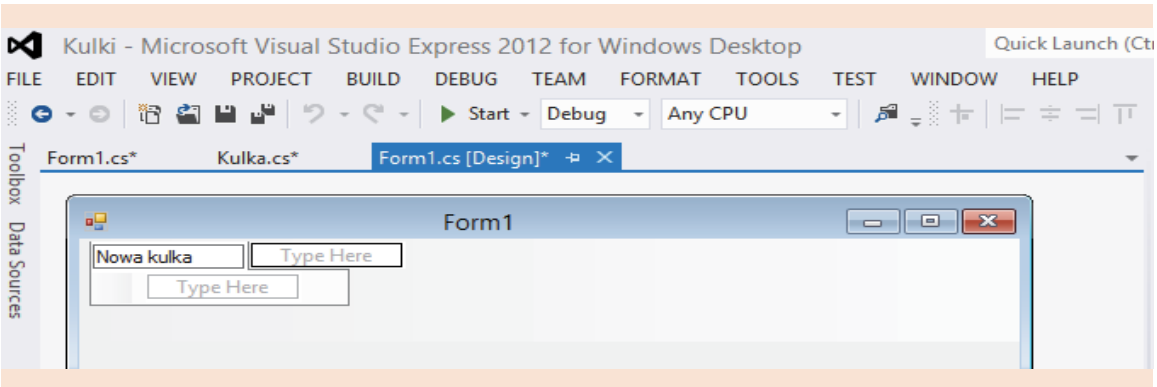
3. Zdefiniowanie na formularzu menu z opcją Nowa kulka.

W kolejnym kroku przygotujemy prezentację tworzonych kulek na formularzu. W tym celu dodajemy do formularze obiekt menu, jak pokazano na rysunku 8.



Rysunek 8. Dodawanie do formularza obiektu MenuStrip.

W pierwsze pole menu wpisujemy tekst Nowa kulka (rysunek 9).



Rysunek 9. Konfigurowanie obiektu MenuStrip.

Ostatnim krokiem, który zostanie zrealizowany w ramach tej lekcji jest napisanie kodu, który będzie się wykonywał, gdy klikniemy na opcję Nowa kulka. Kod obsługi tego zdarzenia pokazano na rysunku 10.

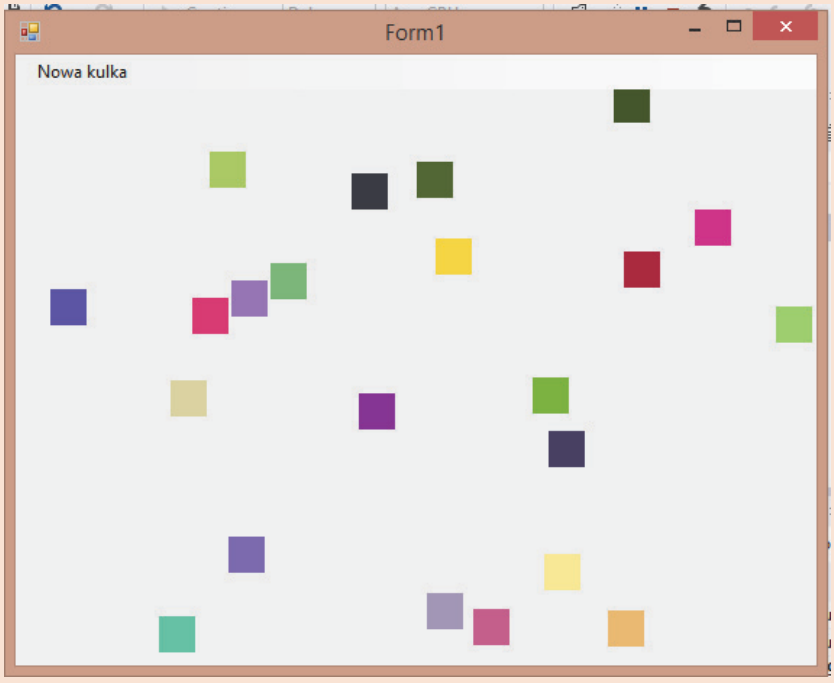
```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }

    private void nowaKulkaToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Kulka tmp = new Kulka();
        tmp.Parent = this;
        tmp.Left = tmp.los.Next(ClientRectangle.Width - tmp.Width);
        tmp.Top = tmp.los.Next(ClientRectangle.Height - tmp.Top);
    }
}
```

Rysunek 10. Kod obsługi opcji Nowa Kulka.

Po wykonaniu ostatniego zadania program jest gotowy do prezentacji, ale musimy pamiętać, że kulki nie będą się jeszcze przemieszczać na ekranie. Proces definiowania klasy Kulka został szczegółowo omówiony i pokazany w filmie instruktażowym o nazwie Definiowanie klasy w środowisku Visual Studio.

4. Prezentacja działania programu i omówienie zakresu dalszych prac.

Po uruchomieniu programu, każde kliknięcie na opcję Nowa kulka powoduje umieszczenie kulki w losowym miejscu ekranu. Przykładową postać ekranu aplikacji uwidacznia rysunek 11.



Rysunek 11. Przykładowy wygląd ekranu aplikacji.

W ramach kolejnej lekcji zostanie napisany kod metody Lec klasy Kulka co spowoduje, że statyczne kulki pokazane w oknie aplikacji zostaną wprawione w ruch.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom rozszerzony)

Cele kształcenia – wymagania ogólne

I. Rozwiązywanie problemów i podejmowanie decyzji z wykorzystaniem komputera, stosowanie podejścia algorytmicznego.

Treści nauczania – wymagania szczegółowe

Uczeń:

1) analizuje, modeluje i rozwiązuje sytuacje problemowe z różnych dziedzin;
2) stosuje podejście algorytmiczne do rozwiązywania problemu;
3) formułuje przykłady sytuacji problemowych, których rozwiązanie wymaga podejścia algorytmicznego i użycia komputera;
4) dobiera efektywny algorytm do rozwiązania sytuacji problemowej i zapisuje go w wybranej notacji;
5) posługuje się podstawowymi technikami algorytmicznymi;
6) ocenia własności rozwiązania algorytmicznego (komputerowego), np. zgodność ze specyfikacją, efektywność działania;
7) opracowuje i przeprowadza wszystkie etapy prowadzące do otrzymania poprawnego rozwiązania problemu: od sformułowania specyfikacji problemu po testowa nie rozwiązania.

Cel

Podstawowym celem lekcji jest wyjaśnienie podstaw programowania obiektowego. Wyjaśnienie pojęcia klasa i obiekt. Nauczenie podstawowych konstrukcji składniowych.

Słowa kluczowe

klasa, obiekt, konstruktor, metoda, Visual Studio

Co przygotować

- Zainstalować Microsoft Visual Studio Express 2012 for Windows Desktop – oprogramowanie, które jest darmową wersją pakietu Visual Studio 2012, można pobrać pod adresem internetowym:

  http://www.microsoft.com/en-us/download/details.aspx?id=40787

  Nauczyciel może wykorzystać inne środowisko programistyczne, o ile zna je dobrze. Wprowadzenie do cyklu lekcji scenariusza omówić z wykorzystaniem prezentacji, zawartej w plikach dodatkowych, o nazwie „Co to znaczy obiektowo”

Przebieg zajęć

Wprowadzenie (15 minut)

W trakcie wprowadzenia wykorzystujemy prezentację o nazwie „Co to znaczy obiektowo”.

1. Na początku zajęć nauczyciel podaje temat lekcji „Klasa to podstawa”, omawia krótko istotę hermetyzacji w programowaniu obiektowym i przystępuje do wyjaśnienia pojęcia klasy i obiektu.

2. Po omówieniu podstaw, nauczyciel przystępuje do opisu zadania, które będzie realizowane, zwracając szczególną uwagę na fakt, że do jego pełnej realizacji wystarczy zdefiniowanie jednej klasy.
3. Na zakończenie tej części lekcji omawia sposób uruchomienia środowiska Visual Studio 2012 i przystępuje do realizacji zadania. Szczegóły zadania opisane zostały w streszczeniu do lekcji.

Zasadnicza część lekcji (25 minut)

W zasadniczej części lekcji wspólnie z uczniami definiowana jest klasa o nazwie Kulka oraz opracowany zostanie program prezentujący tworzenie obiektów klasy Kulka i prezentowanie ich na ekranie. Należy wyraźnie zaznaczyć, że zadania wykonywane w ramach lekcji stanowią wstęp do dalszej pracy i przygotowany przykład klasy i jej zastosowania zostanie zmodyfikowany na kolejnych zajęciach.

Podsumowanie (5 minut)

W ramach dyskusji należy zwrócić szczególną uwagę na fakt, że jedna definicja klasy pozwala na tworzenie dowolnej liczby obiektów, które mogą różnić się pewnymi cechami.

Sprawdzenie wiedzy

Podstawowe sprawdzenie wiedzy można zrealizować na podstawie testu o nazwie test1.docx.

Ocenianie

Oceny na podstawie testu oraz aktywności uczniów w trakcie zajęć.

Dostępne pliki

1. Prezentacja z opisem procesu normalizacji – Co to znaczy obiektowo.pptx
2. Film instruktażowy 1 – Definiowanie klasy w środowisku Visual Studio
3. Test 1

LEKCJA NR 2

TEMAT: Symulacja ruchu, czyli szaleństwo na ekranie

Streszczenie

Zadaniem, które będziemy realizować w ramach tej lekcji to modyfikacja deklaracji klasy kulka w celu symulacji jej ruchu oraz odbić od ścian okna i innych kulek. Efekt ruchu osiągniemy dzięki umieszczonemu w klasie Kulka obiekcie Silnik klasy Timer. Klasa Timer istotny element: właściwość Interval, w której określamy, co ile milisekund ma być generowane zdarzenie o nazwie Tick. Naszym zadaniem będzie napisanie kodu metody Lec klasy Kulka oraz dowiązanie tej metody do zdarzenia Tick klasy Timer. Efektem zmian będzie automatyczne wymuszenie co określoną liczbę milisekund wykonania metody obsługującej symulację ruchu kulki.

Na rysunku 12 pokazano modyfikacje deklaracji klasy Kulka po ustaleniu, że metoda Lec będzie uruchamiana jako obsługa zdarzenia o nazwie Tick klasy Timer.

```
namespace Kulki
{
    class Kulka :Panel
    {
        int dx;
        int dy;
        Timer Silnik= new Timer();
        public Random los = new Random();

        public Kulka()
        {
            BackColor=Color.FromArgb(los.Next(255), los.Next(255), los.Next(255));
            dx = 5 - los.Next(10);
            dy = 5 - los.Next(10);
            Width = 25;
            Height = 25;
            Silnik.Interval = 50;
            Silnik.Tick += new EventHandler(lec);
            Silnik.Start();
        }

        void lec(object sender, EventArgs e)
        {


        }
    }
}
```

Rysunek 12. Deklaracja klasy Kulka po zmianach.

W konstruktorze klasy Kulka dodano polecenia inicjujące działanie obiektu klasy Timer:

- Silnik.Interval = 50; - co 50 milisekund będzie generowane zdarzenie Tick klasy Timer,
- Silnik.Tick += new EventHandler(lec); - zdarzenie Tick będzie wykonywało kod metody Lec,
- Silnik.Start(); - inicjalizacja obiektu klasy Timer.

Dodano dwa parametry do metody Lec, jest to wymóg składniowy w sytuacji, gdy metoda będzie dowiązana do obsługi zdarzenia.

Kolejnym krokiem jest napisanie kodu metody Lec. Na rysunku 13 widzimy kod metody Lec realizujący sprawdzenie, czy kulka dotarła do krawędzi okna i w przypadku stwierdzenia tego faktu następuje zmiana kierunku ruchu kulki.

Kod metody Lec realizujący obsługę zderzeń ze ścianami okna pokazano na rysunku 13.

```
        void lec(object sender, EventArgs e)
        {

            if ((Left + dx <= 0) || (Left + Width + dx > Parent.ClientRectangle.Width))
                dx = -dx;
            if ((Top + dy <= 0) || (Top + Height + dy > Parent.ClientRectangle.Height))
                dy = -dy;

            Left += dx;
            Top += dy;

        }
```

Rysunek 13. Kod metody Lec.

Po uruchomieniu programu każda utworzona kulka będzie poruszała się wewnątrz obszaru okna, a po dotarciu do jego krawędzi zmieni kierunek ruchu.

Ostatnie zadanie to napisanie kodu, w metodzie Lec, który będzie identyfikował kolizję z inną kulką i w przypadku kolizji kulki zamienia się właściwościami dx i dy. Kod metody Lec po uzupełnieniu ilustruje rysunek 14.

```
        void lec(object sender, EventArgs e)
        {
            int zamiana;

            if ((Left + dx <= 0) || (Left + Width + dx > Parent.ClientRectangle.Width))
                dx = -dx;
            if ((Top + dy <= 0) || (Top + Height + dy > Parent.ClientRectangle.Height))
                dy = -dy;

            Left += dx;
            Top += dy;

            foreach (Control tmp in Parent.Controls)
            {
                if (tmp is Kulka && tmp != this)
                {
                    if ((tmp.Width / 2 + this.Width / 2) >
                        Math.Sqrt((this.Left - tmp.Left) * (this.Left - tmp.Left) +
                        (this.Top - tmp.Top) * (this.Top - tmp.Top)))
                    {
                        zamiana = this.dx;
                        this.dx = (tmp as Kulka).dx;
                        (tmp as Kulka).dx = zamiana;

                        zamiana = this.dy;
                        this.dy = (tmp as Kulka).dy;
                        (tmp as Kulka).dy = zamiana;

                    };
                };

            };

        }
```

Rysunek 14. Końcowa postać kodu metody Lec.

Po wykonaniu modyfikacji można przystąpić do testowania programu.

Szczegółowe omówienie procesu zmiany deklaracji klasy Kulka zaprezentowano w filmie instruktażowym o nazwie Programowanie ruchu kulki.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom rozszerzony)

Cele kształcenia – wymagania ogólne

I. Rozwiązywanie problemów i podejmowanie decyzji z wykorzystaniem komputera, stosowanie podejścia algorytmicznego.

Treści nauczania – wymagania szczegółowe

Uczeń:

1) analizuje, modeluje i rozwiązuje sytuacje problemowe z różnych dziedzin;
2) stosuje podejście algorytmiczne do rozwiązywania problemu;
3) formułuje przykłady sytuacji problemowych, których rozwiązanie wymaga podejścia algorytmicznego i użycia komputera;
4) dobiera efektywny algorytm do rozwiązania sytuacji problemowej i zapisuje go w wybranej notacji;
5) posługuje się podstawowymi technikami algorytmicznymi;
6) ocenia własności rozwiązania algorytmicznego (komputerowego), np. zgodność ze specyfikacją, efektywność działania;
7) opracowuje i przeprowadza wszystkie etapy prowadzące do otrzymania poprawnego rozwiązania problemu: od sformułowania specyfikacji problemu po testowa nie rozwiązania.

Cel

Podstawowym celem lekcji jest kontynuacja wyjaśnienia podstaw programowania obiektowego. Wyjaśnienie pojęcia zdarzenie i hermetyzacja. Nauczenie podstawowych konstrukcji składniowych.

Słowa kluczowe

zdarzenie, właściwość Parent, klasa Timer

Co przygotować

- W ramach lekcji wykorzystywany jest program, który przygotowany został w trakcie lekcji 1.

Przebieg zajęć

Wprowadzenie (15 minut)

W trakcie wprowadzenia należy przypomnieć i pokazać program, który wykonany został w ramach lekcji 1. Omówić należy istotę zadania, czyli realizację symulacji poruszania się kulek i ich odbić.

Zasadnicza część lekcji (25 minut)

Wspólnie z uczniami omawiamy i wykonujemy zmiany w deklaracji klasy Kulka.

Dyskusja podsumowująca (10 minut)

W ramach dyskusji należy ocenić wykonany program i omówić możliwości jego modyfikacji.

Sprawdzenie wiedzy

W celu sprawdzenia wiedzy można wykorzystać, zamieszczony w materiałach scenariusza test 2.

Ocenianie

Ocena uczniów na podstawie aktywności w trakcie realizowania zadania.

Dostępne pliki

1. Test 2
2. Film instruktażowy 2 – Programowanie ruchu kulki
3. Zadanie

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska
Wyższa Szkoła
Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego