

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** 4

з дисципліни “Основи програмування”

тема “Структури даних”

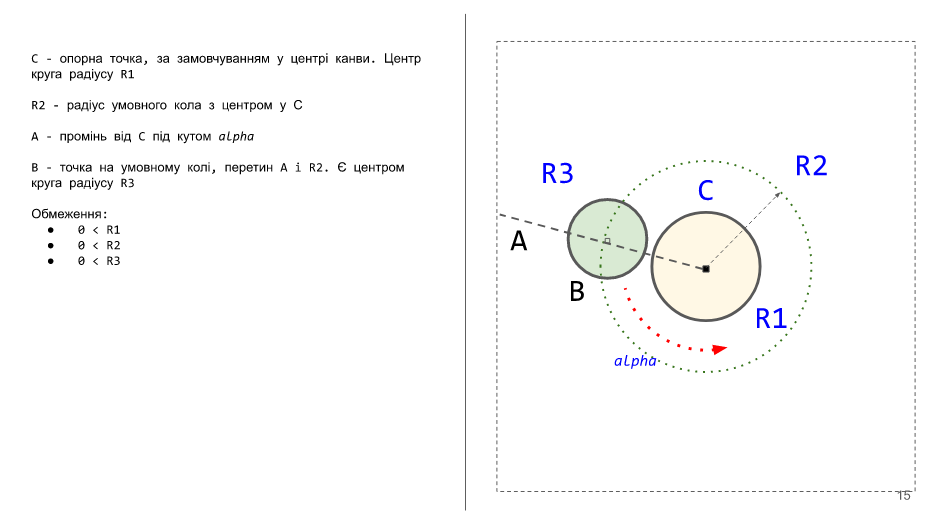
|  |  |  |
| --- | --- | --- |
| Виконав(ла)  студент(ка) I курсу  групи КП-01  Беліцький Олександр Сергійович  (*прізвище, ім’я, по батькові*)  варіант № 3 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2020

**Мета роботи**

Навчитися створювати нові типи даних структур та використовувати змінні цих типів.  
Навчитись виконувати операції над значеннями структур даних та використовувати значення-структури у користувацьких функціях. Навчитися використовувати масиви структур даних, посилання на структури та масиви структур даних у користувацьких функціях.

**Постановка завдання**

Намалювати у консолі набір графічних взаємозалежних об'єктів, параметри яких можна змінювати вводом користувача.

**Аналіз вимог і проектування**

Графічні об'єкти, імена яких виділені на слайді варіанту синім кольором можна змінювати командами користувача. Графічні об'єкти з чорними назвами залежать від інших графічних об'єктів і їх параметри потрібно автоматично обчислювати на основі параметрів інших об'єктів.

Варіант завдання містить обмеження деяких параметрів. При зміні параметрів не допускати вихід їх значень за допустимі межі.

Графічні об`єкти:

* Основний круг з центром у точці С і радіусом r1;
* Умовне коло з центром у точці С і радіусом r2;
* Промінь А від точки С під кутом альфа;
* Залежний круг з центром у точці В на перетині умовного кола і променя А.

Обмеження:



У моєму варіанті найдоцільніше використовувати структуру координат декартової системи:

|  |
| --- |
| struct Point  {  public double x;  public double y;  } |

Формули, за якими обчислюються параметри похідних графічних об'єктів:

Залежний круг радіуса r3 із центром в точці В:

|  |
| --- |
| b.x = r2 \* Math.Cos(alpha) + c.x;  b.y = r2 \* Math.Sin(alpha) + c.y;  //малюнок  Canvas.SetColor(150, 255, 150);  Canvas.FillCircle((int)c.x, (int)c.y, (int)r1);  Canvas.SetColor(255, 0, 255);  Canvas.PutPixel((int)c.x, (int)c.y); |

**Тексти коду програм**

|  |
| --- |
| **Program.cs** |
| using System;  using Progbase.Procedural;  using static System.Console;  namespace lab4  {  struct Point  {  public double x;  public double y;  }  class Program  {  static void Main(string[] args)  {  const int size = 40;  Canvas.SetSize(size, size);  Point c = new Point();  c.x = size / 2;  c.y = size / 2;  double r1 = 5;  double r2 = 15;  double r3 = 3;  double alpha = Math.PI / 3;  Point b = new Point();  Canvas.InvertYOrientation();  ConsoleKeyInfo keyInfo;  Console.Clear();  do  {  b.x = r2 \* Math.Cos(alpha) + c.x;  b.y = r2 \* Math.Sin(alpha) + c.y;  Canvas.BeginDraw();  Canvas.SetColor(255, 255, 150);  Canvas.FillCircle((int)b.x, (int)b.y, (int)r3);  Canvas.SetColor(255, 0, 0);  Canvas.PutPixel((int)b.x, (int)b.y);  Canvas.SetColor(150, 255, 150);  Canvas.FillCircle((int)c.x, (int)c.y, (int)r1);  Canvas.SetColor(255, 0, 255);  Canvas.PutPixel((int)c.x, (int)c.y);  Canvas.EndDraw();  keyInfo = Console.ReadKey();  if (keyInfo.Key == ConsoleKey.W)  {  if (c.y < size - 1)  c.y += 1;  }  else if (keyInfo.Key == ConsoleKey.S)  {  if (c.y > 0)  c.y -= 1;  }  else if (keyInfo.Key == ConsoleKey.D)  {  if (c.x < size - 1)  c.x += 1;  }  else if (keyInfo.Key == ConsoleKey.A)  {  if (c.x > 0)  c.x -= 1;  }  else if (keyInfo.Key == ConsoleKey.Z)  {  alpha += Math.PI / 10;  }  else if (keyInfo.Key == ConsoleKey.X)  {  alpha -= Math.PI / 10;  }  else if (keyInfo.Key == ConsoleKey.T)  {  r3 += 1;  }  else if (keyInfo.Key == ConsoleKey.Y)  {  if (r3 > 1)  {  r3 -= 1;  }  }  else if (keyInfo.Key == ConsoleKey.G)  {  r2 += 1;  }  else if (keyInfo.Key == ConsoleKey.H)  {  if (r2 > 1)  {  r2 -= 1;  }  }  else if (keyInfo.Key == ConsoleKey.B)  {  r1 += 1;  }  else if (keyInfo.Key == ConsoleKey.N)  {  if (r1 > 1)  {  r1 -= 1;  }  }  } while (keyInfo.Key != ConsoleKey.Escape);  Console.WriteLine();  }  }  } |

**Приклади результатів**

Кнопки-контролери:

W - рух точки С (центр основного круга) вверх

S - рух точки С (центр основного круга) вниз

A - рух точки С (центр основного круга) вліво

D - рух точки С (центр основного круга) вправо

Z - збільшення кута альфа (поворот залежного круга)

X - зменшення кута альфа (поворот залежного круга)

T - збільшення радіуса залежного круга

Y - зменшення радіуса залежного круга

G - збільшення відстані між основним і залежним кругом

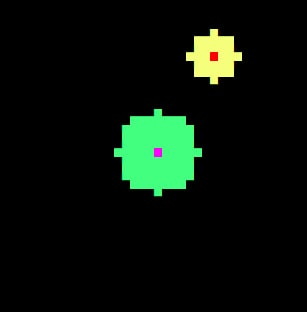
H - зменшення відстані між основним і залежним кругом

B - збільшення радіуса основного круга

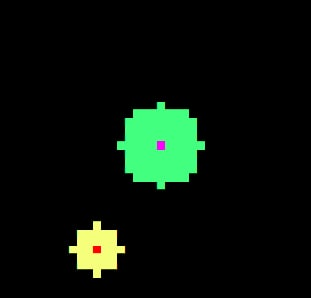
N - зменшення радіуса основного круга

ESC -завершення програми

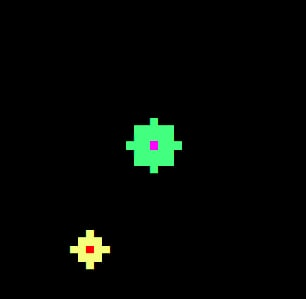
Вихідне положення:

****

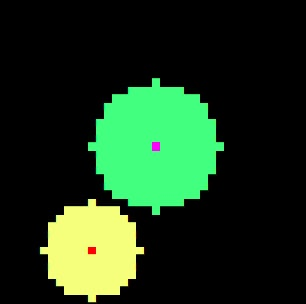
Поворот залежного круга:



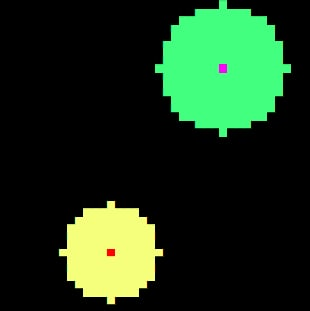
Зменшення радіуса залежного і основного круга:



Збільшення радіуса основного і залежного круга:



Збільшення відстані між кругами:



**Висновки**

Поряд з класами структури представляють ще один спосіб створення власних типів даних в C #. Більш того багато примітивних типів, наприклад, int, double і т.д., по суті є структурами. Тип структури - це тип значення, який може инкапсулювати дані і пов'язані функції. Для визначення типу структури використовується ключове слово ***struct***.

Структури - типи значень, що розміщуються в стеці. З міркувань продуктивності слід використовувати структури для невеликих типів даних. Структура дозволяє об'єднувати декілька різних за типом елементів під одним ім'ям. Більш того можна використовувати неодноразове вкладення одних типів даних в інші, використовуючи структури.

Малювати у терміналі можна за допомогою функцій NuGet пакету ***[Progbase.Canvas](https://www.nuget.org/packages/Progbase.Canvas/)***.Компіляція всього коду відбувалася за допомогою утиліти dotnet.