**«Платформа Microsoft .NET и язык программирования C#» №6**

**Тема: Коллекции. Generic**

**Цель:**

Закрепить у слушателей практические навыки и теоретические знания для работы с коллекциями. Научиться использовать принципы объектно-ориентированного программирования.

**Необходимые инструменты:** MS Visual Studio 2013

**Документация:** Конспект, Литература

**Задание 1. Программа «Карточная игра!»**

Создать модель карточной игры.

Требования:

1. Класс **Game** формирует и обеспечивает:
   * 1. Список игроков (минимум 2);
     2. Колоду карт (36 карт);
     3. Перетасовку карт (случайным образом);
     4. Раздачу карт игрокам (равными частями каждому игроку);
     5. Игровой процесс. **Принцип**: Игроки кладут по одной карте. У кого карта больше, то тот игрок забирает все карты и кладет их в конец своей колоды. Упрощение: при совпадении карт забирает первый игрок, шестерка не забирает туза. Выигрывает игрок, который забрал все карты.
2. Класс **Player** (набор имеющихся карт, вывод имеющихся карт).
3. Класс **Karta** (масть и тип карты (6-10, валет, дама, король, туз).

**Результат решения:**

using System;

using System.Collections.Generic;

namespace test11

{

class Card

{

string name;

char mast;

int point;

string whose;

public string Name

{

get { return name; }

set { name = value; }

}

public char Mast

{

get { return mast; }

set { mast = value; }

}

public string Whose

{

get { return whose; }

set { whose = value; }

}

public int Point

{

get { return point; }

}

public Card(string name, char mast,int point, string whose)

{

this.name = name;

this.mast = mast;

this.point = point;

this.whose = ".";

this.whose = whose;

}

public void PrintCard()

{

Console.Write(name + mast);

}

}

class Player

{

string name;

List<Card> playerkoloda;

public int GetSizeKoloda()

{

return playerkoloda.Count;

}

public Player(string name)

{

this.name = name;

playerkoloda = new List<Card>();

}

public void MarkCards()

{

for(int i=0;i<playerkoloda.Count;i++)

{

playerkoloda[i].Whose = name;

}

}

public string Name

{

get { return name; }

set { name = value; }

}

public void AddCard(Card card)

{

playerkoloda.Add(card);

}

public void PrintPlayerKoloda()

{

for(int i=0; i<playerkoloda.Count;i++)

{

playerkoloda[i].PrintCard();

Console.Write(" ");

}

Console.WriteLine();

}

//int GetPoint(string n)

//{

// if (string.Compare(n, "6") == 0)

// return 6;

// if (string.Compare(n, "7") == 0)

// return 7;

// if (string.Compare(n, "8") == 0)

// return 8;

// if (string.Compare(n, "9") == 0)

// return 9;

// if (string.Compare(n, "10") == 0)

// return 10;

// if (string.Compare(n, "J") == 0)

// return 11;

// if (string.Compare(n, "Q") == 0)

// return 12;

// if (string.Compare(n, "K") == 0)

// return 13;

// if (string.Compare(n, "A") == 0)

// return 14;

// return 0;

//}

public Card DeleteAndReturnCard()

{

for (int i = 0; i < playerkoloda.Count; i++)

{

playerkoloda[i].PrintCard();

Console.WriteLine(" ->" + (i + 1));

}

Console.Write("->");

int k = Convert.ToInt32(Console.ReadLine());

Card cardtoreturn = new Card(playerkoloda[k - 1].Name, playerkoloda[k - 1].Mast, playerkoloda[k - 1].Point, playerkoloda[k - 1].Whose/\*GetPoint(playerkoloda[k - 1].Name)\*/);

playerkoloda.RemoveAt(k - 1);

Console.Clear();

return cardtoreturn;

}

}

class Game

{

List<Card> pole;

List<Card> generalkoloda;

List<Player> players;

public Game()

{

pole = new List<Card>();

players = new List<Player>();

generalkoloda = new List<Card>();

generalkoloda.Add(new Card("6", (char)3,6,"."));

generalkoloda.Add(new Card("7", (char)3,7, "."));

generalkoloda.Add(new Card("8", (char)3,8, "."));

generalkoloda.Add(new Card("9", (char)3,9, "."));

generalkoloda.Add(new Card("10", (char)3,10, "."));

generalkoloda.Add(new Card("J", (char)3,11, "."));

generalkoloda.Add(new Card("Q", (char)3,12, "."));

generalkoloda.Add(new Card("K", (char)3,13, "."));

generalkoloda.Add(new Card("A", (char)3,14, "."));

generalkoloda.Add(new Card("6", (char)4,6, "."));

generalkoloda.Add(new Card("7", (char)4,7, "."));

generalkoloda.Add(new Card("8", (char)4,8, "."));

generalkoloda.Add(new Card("9", (char)4,9, "."));

generalkoloda.Add(new Card("10", (char)4,10, "."));

generalkoloda.Add(new Card("J", (char)4,11, "."));

generalkoloda.Add(new Card("Q", (char)4,12, "."));

generalkoloda.Add(new Card("K", (char)4,13, "."));

generalkoloda.Add(new Card("A", (char)4,14, "."));

generalkoloda.Add(new Card("6", (char)5,6, "."));

generalkoloda.Add(new Card("7", (char)5,7, "."));

generalkoloda.Add(new Card("8", (char)5,8, "."));

generalkoloda.Add(new Card("9", (char)5,9, "."));

generalkoloda.Add(new Card("10", (char)5,10, "."));

generalkoloda.Add(new Card("J", (char)5,11, "."));

generalkoloda.Add(new Card("Q", (char)5,12, "."));

generalkoloda.Add(new Card("K", (char)5,13, "."));

generalkoloda.Add(new Card("A", (char)5,14, "."));

generalkoloda.Add(new Card("6", (char)6,6, "."));

generalkoloda.Add(new Card("7", (char)6,7, "."));

generalkoloda.Add(new Card("8", (char)6,8, "."));

generalkoloda.Add(new Card("9", (char)6,9, "."));

generalkoloda.Add(new Card("10", (char)6,10, "."));

generalkoloda.Add(new Card("J", (char)6,11, "."));

generalkoloda.Add(new Card("Q", (char)6,12, "."));

generalkoloda.Add(new Card("K", (char)6,13, "."));

generalkoloda.Add(new Card("A", (char)6,14, "."));

}

public void AddPlayer(Player player)

{

players.Add(player);

}

void PrintPole()

{

Console.WriteLine("Стiл");

Console.WriteLine("------------------");

for (int i=0;i<pole.Count;i++)

{

pole[i].PrintCard();

Console.Write(" ");

}

Console.WriteLine();

Console.WriteLine("------------------");

}

void MakeAMove(int index)

{

Console.WriteLine(players[index].Name+" робiть хiд:");

pole.Add(players[index].DeleteAndReturnCard());

PrintPole();

}

string WinnerAllGame()

{

for(int i=0;i<players.Count;i++)

{

if(players[i].GetSizeKoloda()==36)

{

return players[i].Name;

}

}

return ".";

}

string GetWinner()

{

int point=0;

string winner="";

for(int i=0;i<pole.Count;i++)

{

if(point<pole[i].Point)

{

point = pole[i].Point;

winner = pole[i].Whose;

}

}

return winner;

}

public void StartGame()

{

Shuffle();

for(int i=0;i<generalkoloda.Count;)

{

for(int j=0;j<players.Count;j++)

{

players[j].AddCard(generalkoloda[i]);

i++;

}

}

while (true)

{

for (int i = 0; i < players.Count; i++)

{

players[i].MarkCards();

}

if (string.Compare(WinnerAllGame(), ".") == 0)

{

for (int i = 0; i < players.Count; i++)

{

MakeAMove(i);

}

string winner = GetWinner();

Console.WriteLine("Забирає " + winner);

for (int i = 0; i < players.Count; i++)

{

if (string.Compare(winner, players[i].Name) == 0)

{

for (int j = 0; j < pole.Count; j++)

{

players[i].AddCard(new Card(pole[j].Name, pole[j].Mast, pole[j].Point,"."));

}

break;

}

}

pole.Clear();

}

else

{

Console.WriteLine(WinnerAllGame() + " виграв!");

break;

}

}

}

void Shuffle()

{

Random rand = new Random(DateTime.Now.Millisecond);

int k = rand.Next(50, 200);

int tmp1, tmp2;

for (int i=0;i<k;i++)

{

tmp1 = rand.Next(0, 36);

tmp2 = rand.Next(0, 36);

Card cardtmp = this.generalkoloda[tmp1];

this.generalkoloda[tmp1] = this.generalkoloda[tmp2];

this.generalkoloda[tmp2] = cardtmp;

}

}

//public void ShowAll()

//{

// for(int i=0; i<generalkoloda.Count;i++)

// {

// if (i % 6 == 0&&i!=0)

// Console.WriteLine();

// generalkoloda[i].PrintCard();

// Console.Write(" ");

// }

// Console.WriteLine();

//}

}

class Program

{

public static void Main()

{

Game g = new Game();

g.AddPlayer(new Player("Vania"));

g.AddPlayer(new Player("Anton"));

g.AddPlayer(new Player("Bogdan"));

g.StartGame();

}

}

}