Лабораторная работа № 15.2

Тема: Разработка программ создания и обработки файлов(Сериализация)

Цель работы: Формирование умений и навыков использования файлов и методов работы с ними. Применения сериализации при работе с данными.

Время на выполнение работы: 2 часа

Этапы работы:

I. Ознакомится с теоретическими сведениями
II. Выполнить задания.
III. Ответить на контрольные вопросы

I Теоретические сведения

Сериализация представляет процесс преобразования какого-либо объекта в поток байтов. После преобразования мы можем этот поток байтов или записать на диск или сохранить его временно в памяти. А при необходимости можно выполнить обратный процесс - десериализацию, то есть получить из потока байтов ранее сохраненный объект.

Атрибут Serializable

Чтобы объект определенного класса можно было сериализовать, надо этот класс пометить атрибутом Serializable:

```
[Serializable]
class Person
{
    public string Name { get; set; }
    public int Year { get; set; }

    public Person(string name, int year)
    {
        Name = name;
        Year = year;
    }
}
```

При отстутствии данного атрибута объект Person не сможет быть сериализован, и при попытке сериализации будет выброшено исключение SerializationException.

Сериализация применяется к свойствам и полям класса. Если мы не хотим, чтобы какое-то поле класса сериализовалось, то мы его помечаем атрибутом NonSerialized:

```
[Serializable]
class Person
{
    public string Name { get; set; }
    public int Year { get; set; }

    [NonSerialized]
    public string accNumber;

    public Person(string name, int year, string acc)
    {
        Name = name;
        Year = year;
        accNumber = acc;
    }
}
```

При наследовании подобного класса, следует учитывать, что атрибут Serializable автоматически не наследуется. И если мы хотим, чтобы производный класс также мог бы быть сериализован, то опять же мы применяем к нему атрибут:

```
[Serializable]
class Worker : Person
```

Формат сериализации

Хотя сериализация представляет собой преобразование объекта в некоторый набор байтов, но в действительности только бинарным форматом она не ограничивается. Итак, в .NET можно использовать следующие форматы:

бинарный

SOAP

xml

JSON

Для каждого формата предусмотрен свой класс: для сериализации в бинарный формат - класс BinaryFormatter, для формата SOAP - класс SoapFormatter, для xml - XmlSerializer, для json - DataContractJsonSerializer.

Пример

Игра миллионер. Ваша задача за 20 ходов заработать n очков. Вам дается m очков на старте(при потере очков проигрыш). Вы можете

- купить акции на определенное количество очков. (на следующий ход они с 60 шансом дадут бонус x4 вложений). А с шансом 20 отнимут столько же очков.(реализовать используя события)

- купить недвижимость на определенное количество очков. Со следующего хода она дает 1% вложений с шансом 85%.

- купить золото на определенное количество очков. Со следующего хода оно дает 0.1% вложений с шансом 100%.

```
using System.Threading.Tasks;
using System.IO;
using System.Runtime.Serialization.Formatters.Binary;

namespace GameLab152
{
    [Serializable]
    ссылок: 11
    class Game
    {

        public event EventHandler<MoneyBursEventArgs> StockBurs;

        ссылка: 1
        public Game(int points, int step):this(0,0,0)
        {
            this.Points = points;
            this.Step = step;
        }

        ссылка: 1
        private Game(int stock, int realty, int gold)
        {
            this.stock = stock;
            this.realty = realty;
            this.gold = gold;
            StockBurs += change;
        }

```

```
31          public int Points
32          {
33              get
34              {
35                  return points;
36              }
37
38              set
39              {
40                  points = value;
41              }
42          }
43
            ссылок: 2
44          public int Step
45          {
46              get
47              {
48                  return step;
49              }
50
51              set
52              {
53                  step = value;
54              }
55          }
56
57
```

```
58          public void CrateStep()
59          {
60              step--;
61              onStartStep();
62          }
            ссылка: 1
63          void onStartStep()
64          {
65              int burst = rnd.Next(0, 100);
66              /*купить акции на определенное количество очков.
67               * (на следующий ход они с 60 шансом дадут бонус x4 вложений).
68               * А с шансом 20 отнимут столько же очков.(реализовать используя события)
69               */
70              if (burst < 20)
71                  StockBurs?.Invoke(this, new MoneyBursEventArgs(-stock));
72              else if
73                  (burst < 80)
74                  StockBurs?.Invoke(this, new MoneyBursEventArgs(stock * 4));
75              /*- купить недвижимость на определенное количество очков.
76               * Со следующего хода она дает 1% вложений с шансом 85%.
77               */
78              if (burst < 85)
79                  points += realty / 100;
80              /*
81               * купить золото на определенное количество очков.
82               * Со следующего хода оно дает 0.1% вложений с шансом 100%.
83               */
84              points = points + gold;
85          }
```

```
86          public bool BuyStock(int count)
87          {
88              if (Points >= count)
89              {
90                  stock += count;
91                  points -= count;
92                  return true;
93              }
94              else
95              {
96                  Console.WriteLine("Покупка не удалась!");
97                  return false;
98              }
99          }
            ссылка: 1
100         public bool BuyGold(int count)
101         {
102             if (Points >= count)
103             {
104                 gold += count;
105                 points -= count;
106                 return true;
107             }
108             else
109             {
110                 Console.WriteLine("Покупка не удалась!");
111                 return false;
112             }
113         }
```

```
114         public bool BuyRealty(int count)
115         {
116             if (Points >= count)
117             {
118                 realty += count;
119                 points -= count;
120                 return true;
121             }
122             else
123             {
124                 Console.WriteLine("Покупка не удалась!");
125                 return false;
126             }
127         }
128
            ссылка: 1
129         public void SaveToFile(string path)
130         {
131             BinaryFormatter bf = new BinaryFormatter();
132             using (FileStream fs = new FileStream(path, FileMode.Create))
133             {
134                 bf.Serialize(fs, this);
135             }
136         }
```

```
ссылка: 1
public static Game Load(string path)
{
    BinaryFormatter bf = new BinaryFormatter();
    using (FileStream fs = new FileStream(path, FileMode.Open))
    {
        return (Game)bf.Deserialize(fs);
    }
}
ссылка: 1
void change(object sender, MoneyBursEventArgs burst)
{
    (sender as Game).points += burst.Money;
}
ссылок: 0
public override string ToString()
{
    return $"Ходов осталось {step} Очков {points}\n Золота {gold} Жилья {realty} Акций {stock}";
}
int points;
int step;
int stock, realty, gold;
static Random rnd = new Random();
```

```
class Program
{
    ссылок: 0
    static void Main(string[] args)
    {
        Game game = new Game(100, 20);
        game.StockBurs += Burst;
        while (game.Step>0)
        {
            Menu(ref game);
            if (game.Points <= 0) break;
            if (game.Points >= 1000) break;
        }
        if (game.Points>0)
            Console.WriteLine("Вы выиграли");
        else
            Console.WriteLine("Вы проирали");
        Console.ReadKey();
    }

    ссылка: 1
    private static void Burst(object sender, MoneyBursEventArgs e)
    {
        Console.WriteLine("Произошло изменение на {0}",e.Money);
        Console.WriteLine(sender );
        Console.WriteLine("Стал после");
    }

```

```
ссылка: 1
36      static void Menu(ref Game obj)
37      {
38          Console.WriteLine("Ваш статус:\n" + obj);
39          Console.WriteLine("1 - купить акций");
40          Console.WriteLine("2 - купить недвижимость");
41          Console.WriteLine("3 - купить золото");
42          Console.WriteLine("4 - пропустить ход");
43          Console.WriteLine("5 - сохранить игру");
44          Console.WriteLine("6 - загрузить игру");
45          var choise = int.Parse(Console.ReadLine());
46          switch (choise)
47          {
48              case 1:
49                  SelectCount(obj.BuyStock);break;
50              case 2:
51                  SelectCount(obj.BuyRealty); break;
52              case 3:
53                  SelectCount(obj.BuyGold); break;
54              case 5:
55                  Save(obj); return;
56              case 6:
57                  Load(out obj); return;
58
59              default:
60                  break;
61          }
62          obj.CrateStep();
63
64      }
```

```
ссылок: 4
65      static void SelectCount(Func<int,bool> purchase)
66      {
67          Console.WriteLine("Введите количество");
68          var count = int.Parse(Console.ReadLine());
69          if (!purchase(count))
70          {
71              Console.WriteLine("Вы ввели неправильное количество, повторите ввод");
72              SelectCount(purchase);
73          }
74      }
ссылок: 2
75      static void Load(out Game obj)
76      {
77          Console.WriteLine("Вы пытаетесь загрузить игру");
78          try
79          {
80              obj = Game.Load(SelectPath());
81          }
82          catch (Exception)
83          {
84              Console.WriteLine("Что-то пошло не так повторите ввод!");
85
86              Load(out obj);
87          }
88          Console.WriteLine("Успешно загружен");
89      }
```

```
90          static void Save(Game obj)
91          {
92              Console.WriteLine("Вы пытаетесь Сохранить игру");
93              try
94              {
95                  obj.SaveToFile(SelectPath());
96              }
97              catch (Exception)
98              {
99                  Console.WriteLine("Что-то пошло не так повторите ввод!");
100
101                 Save(obj);
102             }
103             Console.WriteLine("Успешно сохранена");
104         }
            ссылок: 2
105         static string SelectPath()
106         {
107             Console.WriteLine("Введите путь");
108             return Console.ReadLine();
109         }
110
111
112     }
```

II. Выполнить задание

Общее задание: вам необходимо разработать игру. Все взаимодействие осуществляется через меню в консоли. При необходимости вам надо динамически отображать информацию. В любой игре есть опция сохраниться и загрузиться они реализуются через сериализацию(десириализацию). Игры пишутся при помощи классов Игра, Игрок(при необходимости)

Варианты

Вариант 1.

Игра ресторан. Ваша задача за m ходов заработать n очков. Вам дается t очков на старте. Каждый ход в ресторан приходят от 1 до 5 посетителей (Богатый 20%, середнячок 35%, бедный 55%) Вы готовите блюда заранее, каждый берет то блюдо, которое ему по карману, предпочитая самое дорогое из возможных. Если блюдо испортилось генерируется событие и в обработчике его выкидывают за 0.

Суп "Прыжок Будды через стену" стоит 100 очков портится 5 ходов. Дает 250 очков при продаже. Только богатые.

Фриттата с лобстером стоит 50 очков портится 3 хода. Дает 200 очков при продаже. Только богатые

Филе-миньон из вагю стоит 25 портится 7 ходов. Дает 40 при продаже. Покупают богатые и средние клиенты.

Коктейль «Беллини» стоит 40 портится 20 ходов. Дает 50 при продаже. Покупают богатые и средние клиенты.

Кока-кола стоит 15 не портится. Дает 22 при продаже. Покупают все.

Картофель фри стоит 20 портится 1 ход. Дает 30 при продаже. Покупают все.

Вариант 2.

Игра угадай число в игру играет k пользователей по кругу. Программа предлагает пользователю n раз по m попыток.

Ответы программы могут быть **холодно**, когда разница между введенным и задуманным больше 30. **Прохладно** от 30 до 15, Тепло от 15 до 4, Жара если меньше 4х. Каждая из зон ответа должна генерироваться через отдельное событие. Так же генерировать событие при победе конкретного игрока.

Вариант 3.

Игра феодал. Ваша задача накопить n крестьян(если количество крестьян уменьшается меньше 0 игрок проигрывает) при наличии m золота(если золота падает меньше 0 игрок проигрывает). У вас есть k лимит крестьян. Каждый ход добавляется процент крестьян, считается только от целого их количества. Вы можете

- увеличить(уменьшить) налог дает +1/-1 золотой. Уменьшает(увеличивает) процент появления крестьян на 0.2%(можно делать сколько угодна раз в ход)

- Построить лочугу стоит 10 золота. Увеличивает лимит на 4 крестьянина.

- Дать вольную. Уменьшает количество крестьян на 1. Увеличивает процент появления крестьян 5%.

- Провести торжество 20 золота. Увеличивает крестьян на 1го и увеличивает процент появления крестьян на 1%.

Когда появляется крестьянин генерировать событие.

Вариант 4.

Игра танцевальная команда. Задача поднять скил команды на уровень N. Имея M денег. Энергия t(если она падает меньше 0 игрок проигрывает)

- провести занятие. Энергия уменьшается на 10. Скил увеличивается на 2. 100 денег за занятие.

- посмотреть видео с новым танцом. Энергия уменьшится на 2. Скил увеличится на 0.1%(считать только от целых). 5 денег за просмотр.

- провести дуэль (генерируется случайное число в диапозоне +30% -30% от скила команды), если скил исходной команды оказался больше числа, то +20% от скила от числа и + 500 денег, если меньше, то +40% от числа, но -500 денег. Энергия -20.

- отдохнуть за 100 денег энергия равна 100.

Вариант 5.

Игра Ателье. Задача получить n очков, имея m, за t ходов. Если очков меньше 0 игрок проигрывает. Количество посетителей в день p.

- провести рекламу. Стоит 100 очков x2 посетителей.

- снять мерки. Стоит 15 очков +0.25x посетителей.

- Сшить костюм. Указывается количество. Костюмы покупаются не в большем количестве, чем приходят посетители в ход. Стоит 20 очков. При покупке 50 очков.

Случайно может прийти банда в начале хода и украсть все костюмы. Вероятность этого 20%. Генерироваться событие, когда это происходит

III. Контрольные вопросы

1. Сериализация и десериализации.

2. Форматеры для сериализации.

3. Атрибуты используемые в сериализации(десериализации).

4. Граф объектов.

Литература

1. Полный справочник по C#. Г. Шилдт. Издательский дом «Вильямс», 2004.
2. C# в подлиннике. Наиболее полное руководство. Х.Дейтел.
3. C# в задачах и примерах. Культин. Н.Б.
4. C# учебный курс. Г.Шилдт. СПб.: Питер, 2002.
5. C# программирование на языке высокого уровня Павловская Т.А. СПб.: БХВ-Петербург.

Составители ***Чеботарев А.В., Попок О.В.***