# ЛАБОРАТОРНАЯ РАБОТА №4

Курс «Технологии разработки программного обеспечения»

**Тема:** Рефакторинг.

**Цель:** Научиться применять приемы рефакторинга, касающиеся реорганизации функций и данных; реорганизации условных выражений; задач обобщения.

**Задание:**

1. Выполнить рефакторинг кода проекта и создать новую ревизию (или ветку) проекта. Необходимый минимум рефакторингов включает: выделение, встраивание и перемещение метода, встраивание и замена временной переменной, добавление, удаление и замена параметра метода.
2. Выполнить рефакторинг кода проекта и создать новую ревизию (или ветку) проекта. Необходимый минимум рефакторингов включает: декомпозиция условного оператора, консолидация условного выражения, консолидация дублирующихся условных фрагментов, удаление управляющего флага.
3. Выполнить рефакторинг кода проекта и создать новую ревизию (или ветку) проекта. Необходимый минимум рефакторингов включает: подъем и спуск метода, подъем и спуск поля.
4. Выполнить рефакторинг кода, приведенного в приложении А. Продемонстрировать журнал ревизий и привести листинг кода, в котором был произведен рефакторинг.

**Контрольные вопросы:**

1. Приведите определения рефакторинга. Что отличает рефакторинг от оптимизации производительности?
2. Укажите цели рефакторинга.
3. Когда следует проводить рефакторинг и когда он не нужен?
4. Приведите список признаков проблемного кода («запахи» кода).
5. Перечислите приемы рефакторинга, касающиеся составления методов. Опишите приемы «выделение метода» и «встраивание метода».
6. Перечислите приемы рефакторинга, касающиеся перемещения функций между объектами. Опишите приемы «перемещение метода» и «перемещение поля».
7. Перечислите приемы рефакторинга, касающиеся реорганизации данных.
8. Перечислите приемы рефакторинга, касающиеся упрощения вызовов методов.
9. Перечислите приемы рефакторинга, касающиеся реорганизации условных выражений.
10. Перечислите приемы рефакторинга, касающиеся решения задач обобщения.

**Рекомендуемые источники.**


1. Фаулер, М. Рефакторинг: улучшение существующего кода. – СПб.: Символ-плюс, 2003. – 432с.
2. Мартин Р. Чистый код: Создание, анализ и рефакторинг. Библиотека программиста. – СПб.: Питер, 2010. – 464с.
3. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж.. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб.: Питер, 2001. – 368с.

**Приложение А. Фрагменты кода, подлежащие рефакторингу (взяты с сайта govnokod.ru).**

1) ----------

```
private void set_mode(bool mod)
{
    if(mod==true)
    {
        label1.Enabled=true;
        button1.Enabled=true;
        button2.Enabled=true;
        button3.Enabled=false;
        button4.Enabled=false;
    }
    else
    {
        label1.Enabled=false;
        button1.Enabled=false;
        button2.Enabled=false;
        button3.Enabled=true;
        button4.Enabled=true;
    }
}
```

2) ----------

```
switch (driver.Status)
{
  case ClientStatus.Unknown:
    return m_driverStatusNames[ClientStatus.Unknown];
  case ClientStatus.Free:
    return m_driverStatusNames[ClientStatus.Free];
  case ClientStatus.Busy:
    return m_driverStatusNames[ClientStatus.Busy];
  case ClientStatus.InWay:
    return m_driverStatusNames[ClientStatus.InWay];
  case ClientStatus.Work:
    return m_driverStatusNames[ClientStatus.Work];
  case ClientStatus.Break:
    return m_driverStatusNames[ClientStatus.Break];
  case ClientStatus.Alarm:
    return m_driverStatusNames[ClientStatus.Alarm];
}
```

3) ----------

```
uint i;
…
if (i.ToString().Length == 1)
{
  ...
}
```

4) ----------

```
string destination = null;
for (int i = 0; i < 13; i++)
    destination += source[i];
```

5) ----------

```
Bool IsNumber (string str) {
return (str.Replace ("0", "").Replace ("1", "").Replace ("2", "").Replace
("3", "").Replace ("4", "").Replace ("5", "").Replace ("6", "").Replace
("7", "").Replace ("8", "").Replace ("9", "").Length == 0);
}
```

6) ----------

```
foreach (DirectoryInfo dir in dirs.GetDirectories())
{
      //create folder{16}
      stream.Write(new byte[] { (byte)NetworkMessage.MakeDir }, 0, 1);
      stream.Read(new byte[1],0,  1);

      stream.Write(BitConverter.GetBytes(Encoding.UTF8.GetBytes(
      SubFolder.Replace('\\', '/') + dir.Name.Replace('\\', '/')).Length), 0,  4);

      stream.Write(Encoding.UTF8.GetBytes(SubFolder.Replace('\\', '/') +
         dir.Name.Replace('\\', '/')),0,
                  Encoding.UTF8.GetBytes(SubFolder.Replace('\\', '/') +
         dir.Name.Replace('\\', '/')).Length);
         //send folder name

      stream.Read(new byte[1], 0, 1);//Ok
}
```

7) ----------

```
String[] days = new String[7];
for( int i = 0; i < 7; i++ ) {
  switch(i) {
    default:
    case 0:
      days[i] = "Monday";
      break;
    case 1:
      days[i] = "Tuesday";
      break;
    case 2:
      days[i] = "Wednesday";
      break;
    case 3:
      days[i] = "Thursday";
      break;
    case 4:
      days[i] = "Friday";
      break;
    case 5:
      days[i] = "Saturday";
      break;
    case 6:
      days[i] = "Sunday";
      break;
  }
}
```

8) ----------

```
DateTime dt = DateTime.Now;
string h=dt.Hour.ToString().PadLeft(2,'0');
string m=dt.Minute.ToString().PadLeft(2,'0');
string s=dt.Second.ToString().PadLeft(2,'0');
Console.WriteLine("--"+h+":"+m+":"+s+"--");
```

9) ----------

```
return ((int)(Counter / 2) != Counter / 2.00 && Counter != 0);
```

10) ----------

```
if (Connected == 0))
{
        rez = setup();
        fl_end = true; // выход
}
else
        fl_end = true;
```

11) ----------

```
List<int> arr = new List<int>();
List<int> tmpArr = new List<int>();

for (int i = 0; i < arr.Count; i++)
{
        if (arr[i] > 100)
        {
        }
        else
                tmpArr.Add(arr[i]);
}
arr = tmpArr;
```

12) ----------

```
var ids = form.Keys;

if(ids.Length == 0 || ids.Length > 1) { throw Exception;}
```

13) ----------

```
//Use the Name field
string[] nameParts = customer.Name.Split(' ');

string firstName = nameParts[0];
string lastName = customer.Name.Replace(nameParts[0], "").TrimStart(' ');
```

14) ----------

```
foreach (string id in sourceIDs.Split(new char[] { ',' },
StringSplitOptions.RemoveEmptyEntries))
            {
                sourceId = Convert.ToInt32(id);
                break;
            }
```

15) ----------

```
            public string generateEMail()
            {
                    string res;
                    int i = PersonName.IndexOf(" ");
                    char[] str1 = new char[i];
                    PersonName.CopyTo(0, str1, 0, i);
                    string str11 = new string(str1);
                    char[] str2 = new char[PersonName.Length - i - 1];
                    PersonName.CopyTo(i + 1, str2, 0, PersonName.Length-i-1);
                    string str22 = new string(str2);
                    res = str11.ToString() + "." + str22.ToString();
                    if (res.Length > 20)
                    {
                            str1 = new char[20];
                            res.CopyTo(0, str1, 0, 20);
                            res = new string(str1);
                    }
                    res += "@domain.ua";
                    return res;
            }
```

16) ----------

```
    foreach (string id in sourceIDs.Split(new char[] { ',' },
                            StringSplitOptions.RemoveEmptyEntries))
                {
                    sourceId = Convert.ToInt32(id);
                    break;
                }
```

17) ----------

```
        string GetTextDiv2(string text)
            {
                int mid = text.Length / 2;
                int r = text.IndexOf(" ", mid); if (r < 0) r = 5000;
                int l = text.IndexOf(" ", 0, mid); if (l < 0) l = 5000;
                if (r - mid > mid - l) // to left is closer
                    mid = l;
                else mid = r;

                if (mid == 5000) return " " + text;
                return " " + text.Substring(0, mid) + " <br/> " +
        text.Substring(mid, text.Length - mid);
            }
```

18) ----------

```
private static readonly char SPECIFIER = "$"[0];
private static readonly char DELIMITER = ":"[0];
private static readonly char[] DELIMITER_ARRAY = new char[1] { DELIMITER };
```

19) ----------

```
string mailTo = ((Config.GetSetting("AdminNotifications_EmailAddress") == null) ||
 (Config.GetSetting("AdminNotifications_EmailAddress").Length <= 0))?
Globals.GetHostPortalSettings().HostSettings["SMTPPassword"].ToString():
Config.GetSetting("AdminNotifications_EmailAddress");
```

20) ----------

```
public bool CheckPath(string path)
{
    int n;

    n = 0;
    //Проверяем наличие нужных папок;
    if (Directory.Exists(path + "SCLAD"))
    {
        n += 1;
    }
    if (Directory.Exists(path + "REAL"))
    {
        n += 1;
    }
    if (Directory.Exists(path + "DOSTAVKA"))
    {
        n += 1;
    }
    //Проверяем наличие нужных файлов
    if (File.Exists(path + "analit.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "partner.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mdoc.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mdoc.fpt"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mdocm.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mgrup.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mlabel.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "SCLAD\\mlabel.fpt"))
    {
        n += 1;
    }
    if (File.Exists(path + "REAL\\rbookm.dbf"))
    {
        n += 1;
    }
    if (File.Exists(path + "REAL\\rbook.dbf"))
    {
        n += 1;
    }
```

```
        if (File.Exists(path + "REAL\\rbook.fpt"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\avt.dbf"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\avtm.dbf"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\avtm.fpt"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\cargo.dbf"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\cargom.dbf"))
        {
            n += 1;
        }
        if (File.Exists(path + "DOSTAVKA\\zamena.dbf"))
        {
            n += 1;
        }

        //Если указанная папка содержит все, что нужно
        if (n == 20)
        {
            return true;
        }

        return false;
    }
```

21) ----------

```
            txtContacts.Text = "";
            bool first = true;

            foreach (string contact in contacts)
            {
                    if (first != true)
                            txtContacts.Text += ";";
                    first = false;

                    txtContacts.Text += contact;
            }
```

22) ----------

```
            if (Game1.clou == true)
                 {Game1.clou = false;}
            else
                 { Game1.clou = true; }
```