Шаг 1. Опредение количества символов встречающихся в тексте

1. Default.aspx

МАМА МЫЛА РА	`<asp:TextBox ID="TextBox1" runat="server" Height="77px" Width="244px" TextMode="MultiLine">МАМА МЫЛА РА</asp:TextBox>`
1	`<asp:Button ID="Button1" runat="server" Text="1" />`
	`<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>`

Default.aspx.cs

```
protected void Button1_Click(object sender, EventArgs e)
 {

  }
```

1. Объявляем массив для хранения букв и 3 хэш-таблицы

```
char[] arr;

Hashtable hash = new Hashtable();
Hashtable hash_per = new Hashtable();
Hashtable ht = new Hashtable() {
    {'a', 0},
    {'b', 0},
    {'c', 0},
    {'d', 0},
    {'e', 0},
    {'f', 0},
    {'g', 0},
```

```
{'h', 0},
{'i', 0},
{'j', 0},
{'k', 0},
{'l', 0},
{'m', 0},
{'n', 0},
{'o', 0},
{'p', 0},
{'q', 0},
{'r', 0},
{'s', 0},
{'t', 0},
{'u', 0},
{'v', 0},
{'w', 0},
{'x', 0},
{'y', 0},
{'z', 0},
{' ', 0},

};
```

2. Преобразование строки в массив

```
string s = TextBox1.Text;
char[] arr = s.ToCharArray();
```

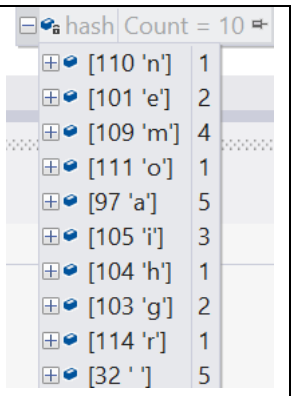
3. Разделение строки на массив букв

mamma mia here i go again	

`int x = 0;`
`foreach (char c in arr)`
`{`
`x = (int)ht[c];`
`x++;`
`ht[c] = x;`
`sa += c + "-" +`
`ht[c] + ", ";`
`}`
	[97 'a'] 5
	[105 'i'] 3
	[113 'q'] 0
	[121 'y'] 0
	[102 'f'] 0
	[110 'n'] 1
	[118 'v'] 0
	[99 'c'] 0
	[107 'k'] 0
	[115 's'] 0
	[104 'h'] 1
	[112 'p'] 0
	[120 'x'] 0
	[101 'e'] 2
	[109 'm'] 4

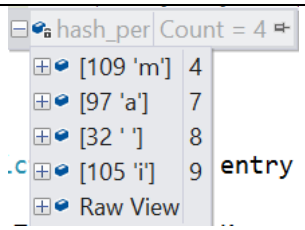
4. Создание короткого хэша hash

```
foreach (DictionaryEntry entry in ht)
{
  if (entry.Value.ToString() != "0")
  {
     sa += entry.Key.ToString() +
       " : " +
       entry.Value.ToString() + ", ";
       hash.Add(entry.Key,entry.Value);
   }
  }
```



5. Создание объекта для хранения пары (буква, %)

```
int sum=0;
foreach (DictionaryEntry entry in
hash)
 {
   sum += (int)entry.Value;
    hash_per.Add(entry.Key,sum);
    }
```



6. Вывод hash_per

```
foreach (DictionaryEntry entry in hash_per)
    {
        Label1.Text += entry.Key + ":" + entry.Value+"<br>";
    }
    int z = hash.Count;
    string u="";
    Label1.Text += "<br>";
```

Number of letters = 10
m:4
a:7
:9
i:10

Написание метода `GenLetter(int n)`

```
internal char GenLetter(int n)
{
    char c=' ';
    foreach (DictionaryEntry entry in hash_per)
    {
        c=(char)entry.Key;
        if (n < (int)entry.Value) return c;
    }
    return c;
}
```

7. Запуск цикла вывода последовательности из 33 символов

```
for (int i = 0; i < 33; i++)
   {
       Label1.Text += ""
             + GenLetter(r.Next(0, num_let+1));
        };
```

Шаг 2. Вычисление нулевой аппроксимации

```
char[] arr0;
internal char GenLetter0(int n)
    {
        return arr0[n];
    }
    protected void Button0_Click(object sender, EventArgs e)
    {
        string sa = "";
        string s = TextBox1.Text;
        char[] arr = s.ToCharArray();

        int x = 0;
```

```
        foreach (char c in arr)
        {
            x = (int)ht[c];
            x++;
            ht[c] = x;
            sa += c + "-" + ht[c] + ", ";

        }

        sa = " ";


        foreach (DictionaryEntry entry in ht)
        {
            if (entry.Value.ToString() != "0")
            {
                sa += entry.Key.ToString() + " : "
                        + entry.Value.ToString() + ", ";
                hash.Add(entry.Key, entry.Value);
            }
        }
        arr0 = new char[hash.Count];
        ICollection keys = hash.Keys;
        Label1.Text = "<font color=red>";
        int j = 0;
        foreach (char c in keys)
        {
            arr0[j] = c;
             j++;
        }
        for (int i = 0; i < 33; i++)
        {

            Label1.Text += "" + GenLetter0(r.Next(0, arr0.Length));
        }

    }
```