

Отчёт к ЛР4

Евгений Черников

1 Алфавитный указатель классов	3
1.1 Классы	3
2 Список файлов	5
2.1 Файлы	5
3 Классы	7
3.1 Класс Gen1	7
3.1.1 Конструктор(ы)	7
3.1.1.1 Gen1()	7
3.1.2 Методы	7
3.1.2.1 gen()	7
3.1.2.2 genR()	8
3.1.2.3 genRVector()	8
3.1.2.4 genVector()	8
3.1.2.5 setSeed()	8
3.2 Класс Gen2	8
3.2.1 Конструктор(ы)	8
3.2.1.1 Gen2()	9
3.2.2 Методы	9
3.2.2.1 gen()	9
3.2.2.2 genR()	9
3.2.2.3 genRVector()	9
3.2.2.4 genVector()	9
3.2.2.5 setSeed()	9
4 Файлы	11
4.1 Файл files.hpp	11
4.1.1 Функции	11
4.1.1.1 fout()	11
4.1.1.2 writeTime()	12
4.1.2 Переменные	12
4.1.2.1 info	12
4.1.2.2 resultFile	12
4.1.2.3 sizes	12
4.1.2.4 sizesNum	12
4.2 files.hpp	12
4.3 Файл gens.hpp	13
4.3.1 Макросы	13
4.3.1.1 ull	13
4.3.2 Функции	13
4.3.2.1 chiSquare()	13
4.3.2.2 D()	14
4.3.2.3 M()	14

---

4.3.2.4 testChi()	14
4.3.2.5 varC()	14
4.3.2.6 writeInfo()	14
4.3.3 Переменные	14
4.3.3.1 module	14
4.4 gens.hpp	15
4.5 Файл main.cpp	16
4.5.1 Функции	17
4.5.1.1 main()	17
4.5.2 Переменные	17
4.5.2.1 sampleNumber	17
4.5.2.2 sampleSize	17
5 Ссылка на репозиторий	19
6 Графики	21



# Глава 1

## Алфавитный указатель классов

### 1.1 Классы

Классы с их кратким описанием.

Gen1	.....	7
Gen2	.....	8



## Глава 2

# Список файлов

### 2.1 Файлы

Полный список файлов.

<a href="#">files.hpp</a>	11
<a href="#">gens.hpp</a>	13
<a href="#">main.cpp</a>	16





## Глава 3

# Классы

### 3.1 Класс Gen1

```
#include <gens.hpp>
```

#### Открытые члены

- [Gen1](#) ([ull module](#))
- void [setSeed](#) ([ull seed](#))
- [ull gen](#) ()
- double [genR](#) ()
- std::vector< [ull](#) > [genVector](#) (int count)
- std::vector< double > [genRVector](#) (int count)

#### 3.1.1 Конструктор(ы)

##### 3.1.1.1 Gen1()

```
Gen1::Gen1 (  
    ull module ) [inline]
```

#### 3.1.2 Методы

##### 3.1.2.1 gen()

```
ull Gen1::gen ( ) [inline]
```

### 3.1.2.2 genR()

```
double Gen1::genR ( ) [inline]
```

### 3.1.2.3 genRVector()

```
std::vector< double > Gen1::genRVector (
    int count ) [inline]
```

### 3.1.2.4 genVector()

```
std::vector< ull > Gen1::genVector (
    int count ) [inline]
```

### 3.1.2.5 setSeed()

```
void Gen1::setSeed (
    ull seed ) [inline]
```

Объявления и описания членов класса находятся в файле:

- [gens.hpp](#)

## 3.2 Класс Gen2

```
#include <gens.hpp>
```

### Открытые члены

- [Gen2](#) ([ull module](#))
- void [setSeed](#) ([ull seed](#))
- [ull gen](#) ()
- double [genR](#) ()
- std::vector< [ull](#) > [genVector](#) (int count)
- std::vector< double > [genRVector](#) (int count)

### 3.2.1 Конструктор(ы)

### 3.2.1.1 Gen2()

```
Gen2::Gen2 (
    ull module ) [inline]
```

## 3.2.2 Методы

### 3.2.2.1 gen()

```
ull Gen2::gen ( ) [inline]
```

### 3.2.2.2 genR()

```
double Gen2::genR ( ) [inline]
```

### 3.2.2.3 genRVector()

```
std::vector< double > Gen2::genRVector (
    int count ) [inline]
```

### 3.2.2.4 genVector()

```
std::vector< ull > Gen2::genVector (
    int count ) [inline]
```

### 3.2.2.5 setSeed()

```
void Gen2::setSeed (
    ull seed ) [inline]
```

Объявления и описания членов класса находятся в файле:

- [gens.hpp](#)



## Глава 4

# Файлы

### 4.1 Файл files.hpp

```
#include <iostream>
#include <string>
#include <vector>
#include <chrono>
#include <fstream>
```

#### Функции

- std::ofstream fout (resultFile)
- void writeTime (std::string title, std::chrono::steady\_clock::time\_point start, std::chrono::steady\_clock::time\_point end, int divideBy=1)

#### Переменные

- std::string info = "info.txt"
- std::string resultFile = "res.txt"
- int sizesNum = 7
- int sizes [7] = { 1000, 5000, 10000, 50000, 100000, 500000, 1000000 }

#### 4.1.1 Функции

##### 4.1.1.1 fout()

```
std::ofstream fout (
    resultFile )
```

#### 4.1.1.2 writeTime()

```
void writeTime (
    std::string title,
    std::chrono::steady_clock::time_point start,
    std::chrono::steady_clock::time_point end,
    int divideBy = 1 )
```

### 4.1.2 Переменные

#### 4.1.2.1 info

```
std::string info = "info.txt"
```

#### 4.1.2.2 resultFile

```
std::string resultFile = "res.txt"
```

#### 4.1.2.3 sizes

```
int sizes[7] = { 1000, 5000, 10000, 50000, 100000, 500000, 1000000 }
```

#### 4.1.2.4 sizesNum

```
int sizesNum = 7
```

## 4.2 files.hpp

[См. документацию.](#)

```
00001 #ifndef FILES_H
00002 #define FILES_H
00003 #include <iostream>
00004 #include <string>
00005 #include <vector>
00006 #include <chrono>
00007 #include <fstream>
00008
00009 std::string info = "info.txt";
00010 std::string resultFile = "res.txt";
00011
00012 std::ofstream fout(resultFile);
00013
00014 int sizesNum = 7;
00015
00016 int sizes[7] = { 1000, 5000, 10000, 50000, 100000, 500000, 1000000 };
00017
00018 void writeTime(std::string title, std::chrono::steady_clock::time_point start, std::chrono::steady_clock::time_point end,
00019               int divideBy = 1) {
00019     fout << title;
00020
00021     fout << std::chrono::duration_cast<std::chrono::microseconds>(end - start).count() / divideBy << " [микросекунд]\n";
00022 }
00023
00024 #endif // FILES_H
```

## 4.3 Файл gens.hpp

```
#include <vector>
#include <deque>
#include "files.hpp"
```

### Классы

- class [Gen1](#)
- class [Gen2](#)

### Макросы

- `#define ull unsigned long long`

### Функции

- double [M](#) (std::vector< double > &v)
- double [D](#) (double mean, std::vector< double > &v)
- double [varC](#) (double deviation, double mean)
- double [chiSquare](#) (std::vector< double > &v)
- bool [testChi](#) (double chi)
- void [writeInfo](#) (std::vector< double > &vec)

### Переменные

- `const ull module = 1 << 20`

#### 4.3.1 Макросы

##### 4.3.1.1 ull

```
#define ull unsigned long long
```

#### 4.3.2 Функции

##### 4.3.2.1 chiSquare()

```
double chiSquare (
    std::vector< double > & v )
```

#### 4.3.2.2 D()

```
double D (  
    double mean,  
    std::vector< double > & v )
```

#### 4.3.2.3 M()

```
double M (  
    std::vector< double > & v )
```

#### 4.3.2.4 testChi()

```
bool testChi (  
    double chi )
```

#### 4.3.2.5 varC()

```
double varC (  
    double deviation,  
    double mean )
```

#### 4.3.2.6 writeInfo()

```
void writeInfo (  
    std::vector< double > & vec )
```

### 4.3.3 Переменные

#### 4.3.3.1 module

```
const ull module = 1 << 20
```



## 4.4 gens.hpp

[См. документацию.](#)

```

00001 #ifndef GENS_HPP
00002 #define GENS_HPP
00003 #include <vector>
00004 #include <deque>
00005 #include "files.hpp"
00006 #define ull unsigned long long
00007
00008 const ull module = 1 « 20;
00009
00010 class Gen1 {
00011     ull m;
00012     ull a = 1234123421;
00013     ull c = 754342159;
00014     ull k = 7;
00015     ull seed = 0;
00016 public:
00017     Gen1(ull module) {
00018         m=module;
00019     }
00020     void setSeed(ull seed) {
00021         this->seed = seed;
00022     }
00023     ull gen() {
00024         seed = (a^k * seed + (a^k - 1) / (a - 1) * c) % m;
00025         return seed;
00026     }
00027     double genR() {
00028         return (double)gen() / m;
00029     }
00030     std::vector<ull> genVector(int count) {
00031         std::vector<ull> res(count);
00032         for(int i = 0; i < count; ++i)
00033             res[i] = this->gen();
00034         return res;
00035     }
00036     std::vector<double> genRVector(int count) {
00037         std::vector<double> res(count);
00038         for(int i = 0; i < count; ++i)
00039             res[i] = this->genR();
00040         return res;
00041     }
00042 };
00043
00044 class Gen2 {
00045     int r;
00046     int l;
00047     std::deque<ull> nums;
00048     ull m;
00049 public:
00050     Gen2(ull module) {
00051         r=55;
00052         l=24;
00053         m=module;
00054         nums.clear();
00055         Gen1 eng(module);
00056         std::vector<ull> temp = eng.genVector(r);
00057         for(int i = 0; i < r; ++i)
00058             nums.push_back(temp[i]);
00059     }
00060     void setSeed(ull seed) {
00061         nums.clear();
00062         Gen1 eng(m);
00063         eng.setSeed(seed);
00064         std::vector<ull> temp = eng.genVector(r);
00065         for(int i = 0; i < r; ++i)
00066             nums.push_back(temp[i]);
00067     }
00068     ull gen() {
00069         ull newNumber = (nums[r - 1] * 41 + nums[l - 1] * 37) % m;
00070         nums.push_front(newNumber);
00071         nums.pop_back();
00072         return newNumber;
00073     }
00074     double genR() {
00075         return (double)gen() / m;
00076     }
00077     std::vector<ull> genVector(int count) {
00078         std::vector<ull> res(count);
00079         for(int i = 0; i < count; ++i)
00080             res[i] = this->gen();
00081         return res;
00082     }

```

```

00083     std::vector<double> genRVector(int count) {
00084         std::vector<double> res(count);
00085         for(int i = 0; i < count; ++i)
00086             res[i] = this->genR();
00087         return res;
00088     }
00089 };
00090
00091 double M(std::vector<double>& v) {
00092     double sum = 0;
00093
00094     for (auto& n : v)
00095         sum += n;
00096
00097     return sum / (double)(v.size());
00098 }
00099
00100 double D(double mean, std::vector<double>& v) {
00101     double sum = 0;
00102
00103     for (auto& n : v)
00104         sum += (n - mean) * (n - mean);
00105
00106     return sqrt(sum / v.size());
00107 }
00108
00109 double varC(double deviation, double mean) {
00110     return deviation / mean;
00111 }
00112
00113 double chiSquare(std::vector<double>& v) {
00114     const double n = v.size();
00115     const double k = 10;
00116     const double p = 1 / k;
00117
00118     std::vector<double> n_i(k, 0);
00119     for (auto& el : v)
00120         for (int j = 0; j < k; ++j)
00121             if (p * j <= el && el < p * (j + 1))
00122                 ++n_i[j];
00123
00124     double chi = 0;
00125     for (auto& u : n_i)
00126         chi += u*u/p;
00127     return chi / n - n;
00128 }
00129
00130 bool testChi(double chi) {
00131     return 5.899 <= chi && chi <= 11.39;
00132 }
00133
00134 void writeInfo(std::vector<double>& vec) {
00135     double m = M(vec);
00136     double d = D(m, vec);
00137     double vc = varC(d, m);
00138     double chi = chiSquare(vec);
00139
00140     fout << "Среднее: " << m << "\nОтклонение: " << d << "\nКоэффициент вариации: " << vc
00141           << "\nКритерий хи-квадрат: " << chi << "\nИдеальное значение критерия: "
00142           << 8.343 << "\nКритерий " << (testChi(chi) ? "пройден" : "не пройден") << "\n\n";
00143 }
00144
00145
00146 #endif // GENS_HPP

```

## 4.5 Файл main.cpp

```

#include <random>
#include "gens.hpp"

```

### Функции

- int main ()

## Переменные

- `const ull sampleSize = 67`
- `const ull sampleNumber = 10`

### 4.5.1 Функции

#### 4.5.1.1 main()

```
int main ( )
```

### 4.5.2 Переменные

#### 4.5.2.1 sampleNumber

```
const ull sampleNumber = 10
```

#### 4.5.2.2 sampleSize

```
const ull sampleSize = 67
```



## Глава 5

# Ссылка на репозиторий

Ссылка: [https://github.com/EvgenijCS202/MP\\_4](https://github.com/EvgenijCS202/MP_4)



## Глава 6

### Графики



