

Лабораторные работы №6-7

Создано системой Doxygen 1.9.5

1 Лабораторная работа №3	1
2 Алфавитный указатель пространств имен	3
2.1 Пространства имен	3
3 Иерархический список классов	5
3.1 Иерархия классов	5
4 Алфавитный указатель классов	7
4.1 Классы	7
5 Список файлов	9
5.1 Файлы	9
6 Пространства имен	11
6.1 Пространство имен sex	11
6.1.1 Подробное описание	11
6.1.2 Переменные	11
6.1.2.1 man	11
6.1.2.2 woman	11
7 Классы	13
7.1 Класс MGTUU	13
7.1.1 Подробное описание	13
7.1.2 Конструктор(ы)	13
7.1.2.1 MGTUU()	13
7.1.3 Методы	13
7.1.3.1 generate()	13
7.2 Класс MIEM	14
7.2.1 Подробное описание	14
7.2.2 Конструктор(ы)	14
7.2.2.1 MIEM()	15
7.2.3 Методы	15
7.2.3.1 generate()	15
7.3 Класс TicketGenerator	15
7.3.1 Подробное описание	16
7.3.2 Конструктор(ы)	16
7.3.2.1 TicketGenerator()	16
7.3.3 Методы	16
7.3.3.1 generator()	16
7.4 Класс University	17
7.4.1 Подробное описание	17
7.4.2 Методы	17
7.4.2.1 generate()	17
8 Файлы	19

8.1	Файл D:/c++/labs/lab_6_7/classes.cpp	19
8.2	classes.cpp	19
8.3	Файл D:/c++/labs/lab_6_7/classes.h	20
8.4	classes.h	21
8.5	Файл D:/c++/labs/lab_6_7/main.cpp	21
8.5.1	Функции	22
8.5.1.1	check_input()	22
8.5.1.2	generate_ticket()	23
8.5.1.3	main()	23
8.5.1.4	print() [1/2]	23
8.5.1.5	print() [2/2]	24
8.5.1.6	read() [1/2]	24
8.5.1.7	read() [2/2]	24
8.5.1.8	to_lower()	25
8.5.1.9	to_upper()	25
8.5.1.10	usage()	26
8.5.2	Переменные	26
8.5.2.1	day	26
8.5.2.2	month	26
8.5.2.3	university	26
8.5.2.4	vsex	26
8.5.2.5	year	27
8.6	main.cpp	27

Глава 1

Лабораторная работа №3

Программа получает на вход в первой строку название университета ("МИЕМ"/"MG TUU"), во второй пол студента ("man"/"woman"), в третьей год месяц и день рождения через пробел. После чего возвращает уникальный номер студенческого билета для данного студента.

Ввод:

```
miem  
man  
2004 05 08
```

Вывод:

```
820040508234760
```

Для того, чтобы скомпилировать программу, необходимо написать в консоли `make`. Команда `make clean` удаляет объектные файлы, команда `make distclean` удаляет объектные файлы и скомпилированную программу.

Глава 2

Алфавитный указатель пространств имен

2.1 Пространства имен

Полный список пространств имен.

sex	Пространство имен, отвечающее за пол студента. Используется в функциях для создания студ. билета	11
-----	--	----

Глава 3

Иерархический список классов

3.1 Иерархия классов

Иерархия классов.

TicketGenerator	15
University	17
MGTUU	13
MIEM	14

Глава 4

Алфавитный указатель классов

4.1 Классы

Классы с их кратким описанием.

MGTUU	
Является подклассом University (стр. 17). Отвечат за генерацию билета для МГ-ТУУ	13
MIEM	
Является подклассом University (стр. 17). Отвечат за генерацию билета для МИЭМ	14
TicketGenerator	
Вспомогательный класс, для создания студ. билетов	15
University	
Абстрактный класс. Является суперклассом для классов MIEM (стр. 14) и MGTUU (стр. 13)	17

Глава 5

Список файлов

5.1 Файлы

Полный список файлов.

D:/c++/labs/lab_6_7/ classes.cpp	19
D:/c++/labs/lab_6_7/ classes.h	20
D:/c++/labs/lab_6_7/ main.cpp	21

Глава 6

Пространства имен

6.1 Пространство имен sex

Пространство имен, отвечающее за пол студента. Используется в функциях для создания студ. билета.

Переменные

- `int man = 0`
- `int woman = 1`

6.1.1 Подробное описание

Пространство имен, отвечающее за пол студента. Используется в функциях для создания студ. билета.

6.1.2 Переменные

6.1.2.1 man

```
int sex::man = 0
```

См. определение в файле `main.cpp` строка 29

6.1.2.2 woman

```
int sex::woman = 1
```

См. определение в файле `main.cpp` строка 30

Глава 7

Классы

7.1 Класс MGTUU

Является подклассом `University` (стр. 17). Отвечат за генерацию билета для МГТУУ.

```
#include <classes.h>
```

Открытые члены

- `MGTUU()`
- `std::string generate(int, int, int, int) override`
Генерирует номер студ. билета МГТУУ на основе пола и даты рождения студента.

7.1.1 Подробное описание

Является подклассом `University` (стр. 17). Отвечат за генерацию билета для МГТУУ.

См. определение в файле `classes.h` строка 32

7.1.2 Конструктор(ы)

7.1.2.1 MGTUU()

```
MGTUU::MGTUU ( ) [inline]
```

См. определение в файле `classes.h` строка 38

7.1.3 Методы

7.1.3.1 generate()

```
std::string MGTUU::generate (
    int sex,
    int y,
    int m,
    int d ) [override], [virtual]
```

Генерирует номер студ. билета МГТУУ на основе пола и даты рождения студента.

Аргументы

in	sex	- пол (1 - женский, 0 - мужской)
in	y	- год рождения
in	m	- месяц рождения
in	d	- день рождения

Возвращает

ticket - номер студ. билета.

Первая цифра билета зависит от пола. Затем к ней добавляется дата рождения (г, м, д). После чего генерируется 5-значное случайное число. Если число при делении на 10 дает нечетный остаток, то заменяется вторая цифра случайного числа, пока остаток не станет четным. После чего подбирается последняя цифра.

Замещает University (стр. 17).

См. определение в файле classes.cpp строка 15

Объявления и описания членов классов находятся в файлах:

- D:/c++/labs/lab_6_7/ classes.h
- D:/c++/labs/lab_6_7/ classes.cpp

7.2 Класс MIEM

Является подклассом University (стр. 17). Отвечат за генерацию билета для МИЭМ.

```
#include <classes.h>
```

Открытые члены

- MIEM ()
 - std::string generate (int, int, int, int) override
- Генерирует номер студ. билета МИЭМ на основе пола и даты рождения студента.

7.2.1 Подробное описание

Является подклассом University (стр. 17). Отвечат за генерацию билета для МИЭМ.

См. определение в файле classes.h строка 19

7.2.2 Конструктор(ы)

7.2.2.1 MIEM()

```
MIEM::MIEM ( ) [inline]
```

См. определение в файле classes.h строка 25

7.2.3 Методы

7.2.3.1 generate()

```
std::string MIEM::generate (
    int sex,
    int y,
    int m,
    int d ) [override], [virtual]
```

Генерирует номер студ. билета МИЭМ на основе пола и даты рождения студента.

Аргументы

in	sex	- пол (1 - женский, 0 - мужской)
in	y	- год рождения
in	m	- месяц рождения
in	d	- день рождения

Возвращает

ticket - номер студ. билета.

Первая цифра билета зависит от пола. Затем к ней добавляется дата рождения (г, м, д). После чего генерируется 4-хзначное случайное число. Если число при делении на 11 дает остаток 10, то заменяется третья цифра случайного числа, пока остаток равен 10. После чего подбирается последняя цифра.

Замещает University (стр. 17).

См. определение в файле classes.cpp строка 76

Объявления и описания членов классов находятся в файлах:

- D:/c++/labs/lab_6_7/ classes.h
- D:/c++/labs/lab_6_7/ classes.cpp

7.3 Класс TicketGenerator

Вспомогательный класс, для создания студ. билетов.

```
#include <classes.h>
```

Открытые члены

- TicketGenerator ()
- University * generator (std::string)

Возвращает объект класса МІЕМ (стр. 14) или MGTUU (стр. 13), в зависимости от переданного аргумента.

7.3.1 Подробное описание

Вспомогательный класс, для создания студ. билетов.

См. определение в файле classes.h строка 46

7.3.2 Конструктор(ы)

7.3.2.1 TicketGenerator()

`TicketGenerator::TicketGenerator () [inline]`

См. определение в файле classes.h строка 48

7.3.3 Методы

7.3.3.1 generator()

`University * TicketGenerator::generator (`
`std::string name)`

Возвращает объект класса МІЕМ (стр. 14) или MGTUU (стр. 13), в зависимости от переданного аргумента.

Аргументы

in	name	- название университета
----	------	-------------------------

Возвращает

university - объект класса МІЕМ (стр. 14) или MGTUU (стр. 13)

См. определение в файле classes.cpp строка 129

Объявления и описания членов классов находятся в файлах:

- D:/c++/labs/lab_6_7/ classes.h
- D:/c++/labs/lab_6_7/ classes.cpp

7.4 Класс University

Абстрактный класс. Является суперклассом для классов МІЕМ (стр. 14) и MGTUU (стр. 13).

```
#include <classes.h>
```

Открытые члены

- `virtual std::string generate (int, int, int, int)=0`

7.4.1 Подробное описание

Абстрактный класс. Является суперклассом для классов МІЕМ (стр. 14) и MGTUU (стр. 13).

См. определение в файле `classes.h` строка 11

7.4.2 Методы

7.4.2.1 generate()

```
virtual std::string University::generate (  
    int ,  
    int ,  
    int ,  
    int ) [pure virtual]
```

Замещается в МІЕМ (стр. 15) и MGTUU (стр. 13).

Объявления и описания членов класса находятся в файле:

- `D:/c++/labs/lab_6_7/ classes.h`

Глава 8

Файлы

8.1 Файл D:/c++/labs/lab_6_7/classes.cpp

```
#include "classes.h"
```

8.2 classes.cpp

```
См. документацию.
00001 #include "classes.h"
00002
00015 std::string MGTUU::generate(int sex, int y, int m, int d) {
00016     std::string ticket = "";
00017     if (!sex) {
00018         ticket += std::to_string(man);
00019         sum += man;
00020     }
00021     else {
00022         ticket += std::to_string(woman);
00023         sum += woman;
00024     }
00025
00026     ticket += std::to_string(y * 10000 + m * 100 + d);
00027     for (int i = 1; i < ticket.size(); ++i) {
00028         sum += (ticket[i] - '0') * (i + 1);
00029     }
00030
00031     int date = y + m + d;
00032
00033     std::mt19937 gen(date);
00034     std::uniform_int_distribution<> distr(1000, 9999);
00035     std::string id = std::to_string(distr(gen));
00036
00037     for (int i = 0; i < id.size(); ++i)
00038         sum += (id[i] - '0') * (i + 10);
00039
00040     while ((sum % mult) % 2 == 1) {
00041         sum -= id[1] * 11;
00042         int new_d = (id[1] - '0' + 1) % 10;
00043         char new_dc = new_d + '0';
00044         std::cout << "d " << new_d << ' ' << new_dc << std::endl;
00045         id = {id[0], new_dc, id[2], id[3]};
00046         sum += id[1] * 11;
00047         std::cout << "s " << sum << ' ' << sum % 10 << std::endl;
00048     }
00049
00050     ticket += id;
00051
00052     int ost = sum % mult;
00053     for (int i = 0; i < 10; i++) {
00054         if ((ost + (i * 14) % mult) % mult == 0) {
00055             ost = 0;
00056             ticket += std::to_string(i);
```

```

00057         break;
00058     }
00059 }
00060
00061 return ticket;
00062 }
00063
00076 std::string MIEM::generate(int sex, int y, int m, int d) {
00077     std::string ticket = "";
00078     if (!sex) {
00079         ticket += std::to_string(man);
00080         sum += man;
00081     }
00082     else {
00083         ticket += std::to_string(woman);
00084         sum += woman;
00085     }
00086
00087     ticket += std::to_string(y * 10000 + m * 100 + d);
00088     for (int i = 1; i < ticket.size(); ++i) {
00089         sum += (ticket[i] - '0') * (i + 1);
00090     }
00091
00092     int date = y + m + d;
00093
00094     std::mt19937 gen(date);
00095     std::uniform_int_distribution<> distr(10000, 99999);
00096     std::string id = std::to_string(distr(gen));
00097
00098     for (int i = 0; i < id.size(); ++i)
00099         sum += (id[i] - '0') * (i + 10);
00100
00101     while (sum % mult == 4) {
00102         sum -= id[2] * 12;
00103         int new_d = (id[2] - '0' + 1) % 10;
00104         char new_dc = new_d + '0';
00105         id = {id[0], id[1], new_dc, id[3], id[4]};
00106         sum += id[2] * 12;
00107     }
00108
00109     ticket += id;
00110
00111     int ost = sum % mult;
00112     for (int i = 0; i < 10; ++i) {
00113         if ((ost + (i * 15) % mult) % mult == 0) {
00114             ost = 0;
00115             ticket += std::to_string(i);
00116             break;
00117         }
00118     }
00119
00120     return ticket;
00121 }
00122
00123
00129 University* TicketGenerator::generator(std::string name) {
00130     if (name == "MIEM") {
00131         MIEM* university = new MIEM;
00132         return university;
00133     }
00134     MGTUU* university = new MGTUU;
00135     return university;
00136 }

```

8.3 Файл D:/c++/labs/lab_6_7/classes.h

```

#include <iostream>
#include <string>
#include <random>

```

Классы

- class University

Абстрактный класс. Является суперклассом для классов MIEM (стр. 14) и MGTUU (стр. 13).

- class MIEM
Является подклассом University (стр. 17). Отвечат за генерацию билета для МИЭМ.
- class MGTUU
Является подклассом University (стр. 17). Отвечат за генерацию билета для МГТУУ.
- class TicketGenerator
Вспомогательный класс, для создания студ. билетов.

8.4 classes.h

См. документацию.

```

00001 #ifndef UNIVERSITY
00002 #define UNIVERSITY
00003
00004 #include <iostream>
00005 #include <string>
00006 #include <random>
00007
00011 class University {
00012 public:
00013     virtual std::string generate(int, int, int, int) = 0;
00014 };
00015
00019 class MIEM: public University {
00020     int man    = 8;
00021     int woman  = 4;
00022     int sum    = 0;
00023     int mult   = 11;
00024 public:
00025     MIEM() {}
00026     std::string generate(int, int, int, int) override;
00027 };
00028
00032 class MGTUU: public University {
00033     int man    = 2;
00034     int woman  = 1;
00035     int sum    = 0;
00036     int mult   = 10;
00037 public:
00038     MGTUU() {}
00039     std::string generate(int, int, int, int) override;
00040 };
00041
00042
00046 class TicketGenerator {
00047 public:
00048     TicketGenerator() {};
00049     University* generator(std::string);
00050 };
00051
00052 #endif

```

8.5 Файл D:/c++/labs/lab_6_7/main.cpp

```

#include "classes.h"
#include <algorithm>
#include <fstream>
#include <cstring>

```

Пространства имен

- namespace sex
Пространство имен, отвечающее за пол студента. Используется в функциях для создания студ. билета.

Функции

- `void usage ()`
Выводит в консоль справку.
- `char to_lower (char c)`
Возвращает символ, переведенный в нижний регистр.
- `char to_upper (char c)`
Возвращает символ, переведенный в верхний регистр
- `int check_input ()`
Проверяет введенные пользователем данные на корректность.
- `int read ()`
Считывает данные из потока `cin` и записывает их в глобальные переменные.
- `int read (std::string filename)`
Считывает данные из файла и записывает их в глобальные переменные.
- `void print (std::string t)`
Выводит результат в поток `cout`.
- `void print (std::string t, std::string filename)`
Выводит результат в файл.
- `std::string generate_ticket ()`
Генерирует номер билета на основе введенных данных.
- `int main (int argc, char **argv)`
Обрабатывает параметры запуска.

Переменные

- `int sex::man = 0`
- `int sex::woman = 1`
- `std::string university`
- `std::string vsex`
- `int year`
- `int month`
- `int day`

8.5.1 Функции

8.5.1.1 `check_input()`

```
int check_input ( )
```

Проверяет введенные пользователем данные на корректность.

Возвращает

0 - функция выполнена успешно

1 - обнаружена ошибка

См. определение в файле `main.cpp` строка 74

8.5.1.2 generate_ticket()

```
std::string generate_ticket ( )
```

Генерирует номер билета на основе введенных данных.

Возвращает

ticket - номер билета

В функции создается объект класса TicketGenerator (стр. 15), при помощи которого, в зависимости от введенных данных, генерируется номер студ. билета.

См. определение в файле main.cpp строка 166

8.5.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

Обрабатывает параметры запуска.

Аргументы

in	argc	- количество параметров запуска
in	argv	- список параметров запуска

Возвращает

0 - функция выполнена успешно

1 - обнаружена ошибка

В зависимости от введенных при запуске флагов определяет, откуда будут поступать данные, и куда их потом нужно будет вывести. Флаг `-tofile` сообщает, что вывод будет производиться в файл, после него указывается имя файла. Флаг `-fromfile` сообщает, что ввод будет производиться из файла, после него указывается имя файла. Возможна обработка обоих флагов одновременно. Если во время выполнения произошла ошибка, то выведется ошибка, и программа закончит выполнение с кодом 1, иначе с кодом 0.

См. определение в файле main.cpp строка 190

8.5.1.4 print() [1/2]

```
void print (
    std::string t )
```

Выводит результат в поток cout.

Аргументы

in	t	- номер билета, который необходимо вывести
----	---	--

См. определение в файле main.cpp строка 146

8.5.1.5 print() [2/2]

```
void print (
    std::string t,
    std::string filename )
```

Выводит результат в файл.

Аргументы

in	t	- номер билета, который необходимо вывести
in	filename	- имя файла

См. определение в файле main.cpp строка 153

8.5.1.6 read() [1/2]

```
int read ( )
```

Считывает данные из потока cin и записывает их в глобальные переменные.

Возвращает

- 0 - функция выполнена успешно
- 1 - обнаружена ошибка

См. определение в файле main.cpp строка 106

8.5.1.7 read() [2/2]

```
int read (
    std::string filename )
```

Считывает данные из файла и записывает их в глобальные переменные.

Аргументы

in	filename	- имя файла
----	----------	-------------

Возвращает

0 - функция выполнена успешно

1 - обнаружена ошибка

См. определение в файле main.cpp строка 123

8.5.1.8 to_lower()

```
char to_lower (  
            char c )
```

Возвращает символ, переведенный в нижний регистр.

Аргументы

in	c	- символ
----	---	----------

Возвращает

c - символ, переведенный в нижний регистр

См. определение в файле main.cpp строка 60

8.5.1.9 to_upper()

```
char to_upper (  
            char c )
```

Возвращает символ, переведенный в верхний регистр

Аргументы

in	c	- символ
----	---	----------

Возвращает

c - символ, переведенный в верхний регистр

См. определение в файле main.cpp строка 67

8.5.1.10 usage()

```
void usage ( )
```

Выводит в консоль справку.

Функция выводит справку в консоль, если пользователь неверно ввел флаги при запуске программы.

См. определение в файле `main.cpp` строка 45

8.5.2 Переменные

8.5.2.1 day

```
int day
```

См. определение в файле `main.cpp` строка 37

8.5.2.2 month

```
int month
```

См. определение в файле `main.cpp` строка 36

8.5.2.3 university

```
std::string university
```

См. определение в файле `main.cpp` строка 33

8.5.2.4 vsex

```
std::string vsex
```

См. определение в файле `main.cpp` строка 34

8.5.2.5 year

int year

См. определение в файле main.cpp строка 35

8.6 main.cpp

См. документацию.

```

00001 #include "classes.h"
00002 #include <algorithm>
00003 #include <fstream>
00004 #include <cstring>
00005
00028 namespace sex{
00029     int man = 0;
00030     int woman = 1;
00031 };
00032
00033 std::string university; //< Название университета.
00034 std::string vsex; //< Пол студента.
00035 int year; //< Год рождения.
00036 int month; //< Месяц рождения.
00037 int day; //< День рождения.
00038
00045 void usage() {
00046     std::cout << std::endl;
00047     std::cout << "USAGE: ./prog [--tofile <file_name>] [--fromfile <file_name>]" << std::endl;
00048     std::cout << std::endl;
00049     std::cout << "-----Flags meaning-----" << std::endl;
00050     std::cout << "--tofile -----> write output data in file" << std::endl;
00051     std::cout << "--fromfile -----> read input data from file" << std::endl;
00052     std::cout << std::endl;
00053 }
00054
00060 char to_lower(char c) { return std::tolower(c); }
00061
00067 char to_upper(char c) { return std::toupper(c); }
00068
00074 int check_input() {
00075     if (university != "MIEM" && university != "MG TUU") {
00076         std::cerr << "##### Wrong name of university #####\n";
00077         return 1;
00078     }
00079
00080     if (vsex != "man" && vsex != "woman") {
00081         std::cerr << "##### Wrong sex #####\n";
00082         return 1;
00083     }
00084
00085     if (year < 1900 || year > 5000) {
00086         std::cerr << "##### Wrong year of birthday #####\n";
00087         return 1;
00088     }
00089     if (month < 1 || month > 12) {
00090         std::cerr << "##### Wrong month of birthday #####\n";
00091         return 1;
00092     }
00093     if (day < 1 || day > 31) {
00094         std::cerr << "##### Wrong day of birthday #####\n";
00095         return 1;
00096     }
00097
00098     return 0;
00099 }
00100
00106 int read() {
00107     std::cin >> university;
00108     std::transform(university.begin(), university.end(), university.begin(), to_upper);
00109     std::cin >> vsex;
00110     std::transform(vsex.begin(), vsex.end(), vsex.begin(), to_lower);
00111     std::cin >> year >> month >> day;
00112
00113     if (check_input()) return 1;
00114     return 0;
00115 }
00116
00123 int read(std::string filename) {
00124     std::ifstream file (filename);

```

```

00125     if (!file.is_open()) {
00126         std::cerr << "##### Wrong name of file #####\n";
00127         return 1;
00128     }
00129
00130     std::getline(file, university);
00131     std::transform(university.begin(), university.end(), university.begin(), to_upper);
00132     std::getline(file, vsex);
00133     std::transform(vsex.begin(), vsex.end(), vsex.begin(), to_lower);
00134     file >> year >> month >> day;
00135
00136     file.close();
00137
00138     if (check_input()) return 1;
00139     return 0;
00140 }
00141
00146 void print(std::string t) { std::cout << "Ticket is " << t << std::endl; }
00147
00153 void print(std::string t, std::string filename) {
00154     std::ofstream file (filename);
00155     file << "Ticket is " << t << '\n';
00156     file.close();
00157 }
00158
00166 std::string generate_ticket() {
00167     TicketGenerator ticket_generator;
00168     std::string ticket;
00169     if (vsex == "man")
00170         ticket = ticket_generator.generator(university)->generate(sex::man, year, month, day);
00171     else
00172         ticket = ticket_generator.generator(university)->generate(sex::woman, year, month, day);
00173     return ticket;
00174 }
00175
00190 int main(int argc, char ** argv) {
00191     std::string ticket;
00192     switch (argc) {
00193     case 1:
00194         if (read()) exit(1);
00195         ticket = generate_ticket();
00196         print(ticket);
00197         break;
00198     case 3:
00199         if (!strcmp(argv[1], "--fromfile")) {
00200             if (read(argv[2])) exit(1);
00201             ticket = generate_ticket();
00202             print(ticket);
00203         }
00204         else if (!strcmp(argv[1], "--tofile")) {
00205             if (read()) exit(1);
00206             ticket = generate_ticket();
00207             print(ticket, argv[2]);
00208         }
00209         else {
00210             std::cerr << "##### Wrong flags #####\n";
00211             usage();
00212             exit(1);
00213         }
00214         break;
00215     case 5:
00216         if (!strcmp(argv[1], "--tofile") && !strcmp(argv[3], "--fromfile")) {
00217             if (read(argv[4])) exit(1);
00218             ticket = generate_ticket();
00219             print(ticket, argv[2]);
00220         }
00221         else if (!strcmp(argv[3], "--tofile") && !strcmp(argv[1], "--fromfile")) {
00222             if (read(argv[2])) exit(1);
00223             ticket = generate_ticket();
00224             print(ticket, argv[4]);
00225         }
00226         else {
00227             std::cerr << "##### Wrong flags #####\n";
00228             usage();
00229             exit(1);
00230         }
00231         break;
00232     default:
00233         std::cerr << "##### Wrong flags #####\n";
00234         usage();
00235         exit(1);
00236         break;
00237     }
00238
00239     return 0;
00240 }

```