ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ» Московский институт электроники и математики им. А. Н. Тихонова

Отчет по лабораторным работам №6-7 по дисциплине «Языки программирования»

Выполнил: студент группы СКБ221
Саркисянц Юрий Григорьевич

Москва, 2022

Оглавление

Оглавление

README

САРКИСЯНЦ ЮРИЙ ------— СКБ 221 tg: t.me/yur1y_snc

Иерархический список классов

Иерархия классов

Иерархия классов.

Указатель классов

Классы

Классы с их кратким описанием.

Список файлов

Файлы

Полный список файлов.

- **class.cpp**
- **class.h**
- **main.cpp**

Классы

Класс GenOfTicket

Открытые члены

- **GenOfTicket** ()
- **University** * **generate** (std::string, std::string, int, int, int)

Подробное описание

Вспом. класс для генерации билетов, есть метод, который принимает поля, и в завис. от них создает объект и возвращает его

См. определение в файле **class.cpp** строка **159**

Конструктор(ы)

GenOfTicket::GenOfTicket () `[inline]`

См. определение в файле **class.cpp** строка **161**

Методы

University * GenOfTicket::generate (std::string *name*, std::string *sex*, int *year*, int *month*, int *day*)

См. определение в файле **class.cpp** строка **164**

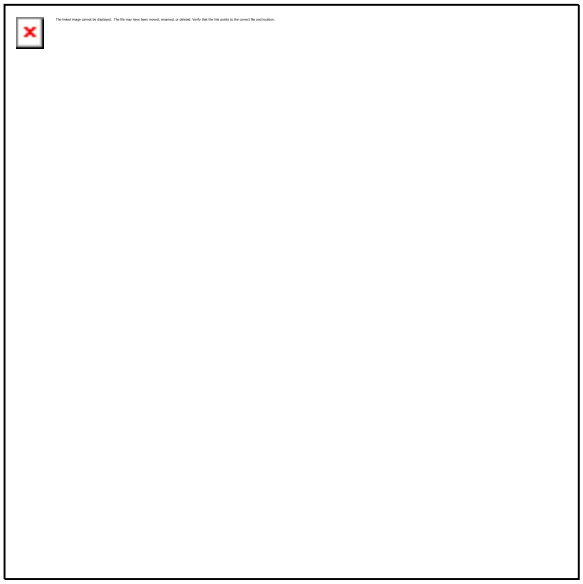
Объявления и описания членов класса находятся в файле:

- **class.cpp**

Класс MGTU

Класс, наследуемый от **University**. Создаем для него конструктор для инициализации полей, и переопределяем метод generator.

Граф наследования:MGTU:



Открытые члены

- **MGTU** (std::string sex, int year, int month, int day)
- std::string **generator** () override
  Генерирует номер студ. билета МГТУ на основе пола и даты рождения студента. Первая цифра билета зависит от пола(0 - мужчина, 1 - женщина). Затем к ней добавляется дата рождения. После чего генерируется 4-хзначное случайное число. Если число при делении на 10 имеет нечетный остаток, то заменяется 3 цифра случайного числа, пока остаток не будет четным. После чего происходит подбор последней цифры.

Подробное описание

Класс, наследуемый от **University**. Создаем для него конструктор для инициализации полей, и переопределяем метод generator.

См. определение в файле **class.cpp** строка **21**

Конструктор(ы)

MGTU::MGTU (std::string *sex*, int *year*, int *month*, int *day*) `[inline]`

См. определение в файле **class.cpp** строка **31**

Методы

std::string MGTU::generator () `[inline]`, `[override]`, `[virtual]`

Генерирует номер студ. билета МГТУ на основе пола и даты рождения студента. Первая цифра билета зависит от пола(0 - мужчина, 1 - женщина). Затем к ней добавляется дата рождения. После чего генерируется 4-хзначное случайное число. Если число при делении на 10 имеет нечетный остаток, то заменяется 3 цифра случайного числа, пока остаток не будет четным. После чего происходит подбор последней цифры.

Замещает **University** (*стр.11*).

См. определение в файле **class.cpp** строка **44**

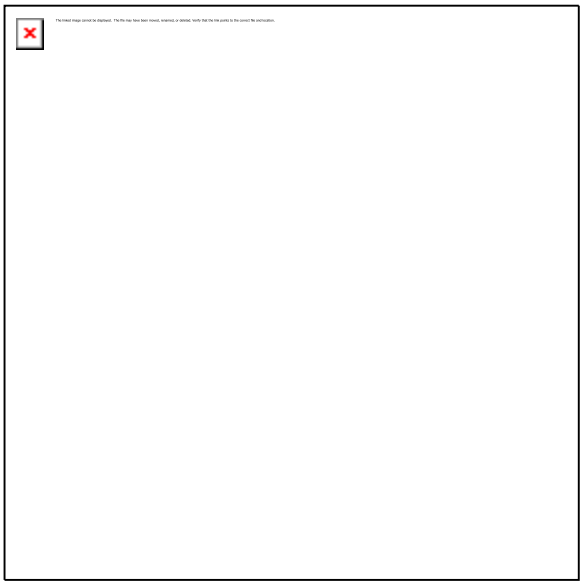
Объявления и описания членов класса находятся в файле:

- **class.cpp**

Класс MIEM

Генерирует номер студ. билета МИЭМА на основе пола и даты рождения студента. Первая цифра билета - это пол(8 мужчина, 4 - женщина. Затем к ней добавляется дата рождения. После чего генерируется 4-хзначное случайное число. Если число при делении на 11 дает остаток 10, то заменяется 3 случайного числа, пока остаток равен 10. После чего происходит подбор последней цифры.

Граф наследования:MIEM:



Открытые члены

- **MIEM** (std::string sex, int year, int month, int day)
- std::string **generator** () override

Подробное описание

Генерирует номер студ. билета МИЭМА на основе пола и даты рождения студента. Первая цифра билета - это пол(8 мужчина, 4 - женщина. Затем к ней добавляется дата рождения. После чего генерируется 4-хзначное случайное число. Если число при делении на 11 дает остаток 10, то заменяется 3 случайного числа, пока остаток равен 10. После чего происходит подбор последней цифры.

См. определение в файле **class.cpp** строка **90**

Конструктор(ы)

MIEM::MIEM (std::string *sex*, int *year*, int *month*, int *day*)`[inline]`

См. определение в файле **class.cpp** строка **100**

Методы

std::string MIEM::generator () `[inline]`, `[override]`, `[virtual]`

Замещает **University** (*стр.11*).

См. определение в файле **class.cpp** строка **107**

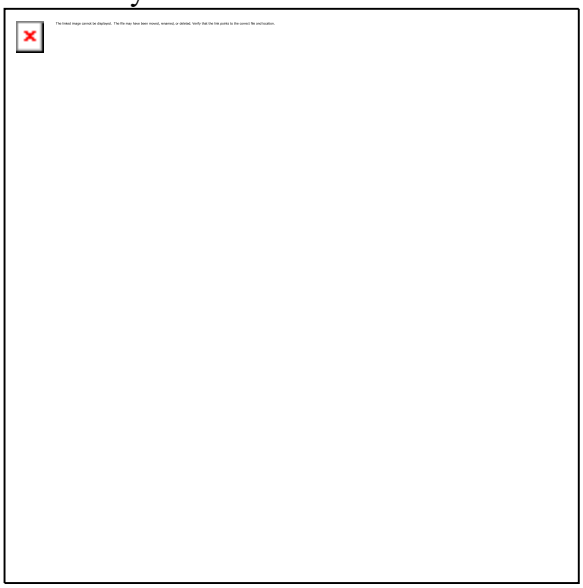
Объявления и описания членов класса находятся в файле:

- **class.cpp**

Класс University

Абстрактный класс. Является родителем для классов **MIEM** и **MGTU**. В нем чисто вирт. функция generator.

Граф наследования:University:



Открытые члены

- virtual std::string **generator** ()=0

Подробное описание

Абстрактный класс. Является родителем для классов **MIEM** и **MGTU**. В нем чисто вирт. функция generator.

См. определение в файле **class.cpp** строка **14**

Методы

virtual std::string University::generator () `[pure virtual]`

Замещается в **MGTU** (*стр.8*) и **MIEM** (*стр.10*).

Объявления и описания членов класса находятся в файле:

- **class.cpp**

Файлы

Файл class.cpp

```
#include <iostream>
#include <time.h>
#include <string>
#include <random>
#include <fstream>
#include <cstring>
```

Классы

- class **University**
  *Абстрактный класс. Является родителем для классов **MIEM** и **MGTU**. В нем чисто вирт. функция generator.*
- class **MGTU**
  *Класс, наследуемый от **University**. Создаем для него конструктор для инициализации полей, и переопределяем метод generator.*
- class **MIEM**
  Генерирует номер студ. билета МИЭМА на основе пола и даты рождения студента. Первая цифра билета - это пол(8 мужчина, 4 - женщина. Затем к ней добавляется дата рождения. После чего генерируется 4-хзначное случайное число. Если число при делении на 11 дает остаток 10, то заменяется 3 случайного числа, пока остаток равен 10. После чего происходит подбор последней цифры.
- class **GenOfTicket**

Функции

- std::string **gen_ticket** ()
- void **FromMenu** ()
- void **ToMenu** ()
- void **FromToMenu** ()
- void **Menu** ()
- void **CheckDate** (int year, int month, int day)
- void **ToFile** (char *namefile)
- void **FromFile** (char *namefile)
- void **FromToFile** (char *namefile1, char *namefile2)

Переменные

- std::string **u_c**
- int **y_c**
- int **m_c**
- int **d_c**
- std::string **s_c**

Функции

void CheckDate (int *year*, int *month*, int *day*)

Функция для проверки даты(мы проверяем, является ли день данного месяца подходящим для него, учитывая высокосность) Если год не высокосный, то в нем в феврале только 28 дней, и в массиве дней месяцев 29 заменяется на 28

См. определение в файле **class.cpp** строка **216**

void FromFile (char * *namefile*)

Функция, принимающая название файла, дальше считывающая данные из файла, введенные через пробел, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в консоль учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **297**

void FromMenu ()

Функция для вывода меню-подсказки при использовании –fromfile

См. определение в файле **class.cpp** строка **188**

void FromToFile (char * *namefile1*, char * *namefile2*)

Функция, принимающая название файла, дальше считывающая данные из файла, введенные через пробел, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в файл учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **360**

void FromToMenu ()

Функция для ввода меню-подсказки при использовании –fromfile и –tofile одновременно

См. определение в файле **class.cpp** строка **200**

std::string gen_ticket ()

См. определение в файле **class.cpp** строка **177**

void Menu ()

Функция для ввода общего меню-подсказки, содержащего выше упомянутые меню

См. определение в файле **class.cpp** строка **206**

void ToFile (char * *namefile*)

Функция, принимающая название файла, дальше требующая данные у пользователя с консоли, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в файл учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **231**

void ToMenu ()

Функция для вывода меню-подсказки при использовании –tofile

См. определение в файле **class.cpp** строка **194**

Переменные

int d_c

См. определение в файле **class.cpp** строка **151**

int m_c

См. определение в файле **class.cpp** строка **150**

std::string s_c

См. определение в файле **class.cpp** строка **152**

std::string u_c

См. определение в файле **class.cpp** строка **148**

int y_c

См. определение в файле **class.cpp** строка **149**

class.cpp

См. документацию.
```
00001 #include <iostream>
00002 #include <time.h>
00003 #include <string>
00004 #include <random>
00005 #include <fstream>
00006 #include <cstring>
00007 using std::cerr;
00008 using std::cout;
00009 using std::endl;
00010 using std::cin;
00014 class University {
00015     public:
00016     virtual std::string generator() = 0;
00017 };
00021 class MGTU: public University
00022 {
00023     int sum = 0;
00024     int div = 10;
00025     std::string sex;
00026     int year;
00027     int month;
00028     int day;
00029     std::string ticket = "";
00030 public:
00031     MGTU(std::string sex,int year, int month, int day)
00032     {
00033         this->sex = sex;
00034         this->year = year;
00035         this->month = month;
00036         this->day = day;
00037     }
00044     std::string generator() override
00045     {
00046         if (sex == "man")
00047         {
00048             ticket+=std::to string(0);
00049             sum+=0;
00050         }
00051         else
00052         {
00053             ticket+=std::to string(1);
00054             sum+=1;
00055         }
00056         int date = 10000*year+100*month+ day;
00057         ticket+=std::to_string(date);
00058         for (int i = 1; i < ticket.size();i++) sum+=(ticket[i] - '0') * (i+1);
00059         std:: mt19937 seq(date + time(0));
00060         std:: uniform int distribution<> gap(1000,9999);
00061         std::string rand number = std::to string(gap(seq));
00062         for (int i = 0; i < rand number.size();i++) sum+=(rand number[i] -
'0')*(10+i);
00063         while (sum % 2 == 1)
00064         {
00065             sum = sum - rand number[2]*(10+2);
00066             int new d = ((rand number[2] - '0')+1)%10 + '0';
00067             rand_number =
{rand number[0],rand number[1],(char)new d,rand number[3]};
00068             sum = sum + new d*(10+2);
00069         }
00070         ticket+=rand number;
00071         int ost = sum / div;
00072         for (int i = 0; i < 10;i++)
00073         {
00074             if ((ost + (14*i)%10)%10 == 0)
00075             {
00076                 ost = 0;
00077                 ticket+=std::to string(i);
00078                 break;
00079             }
00080         }
00081         return ticket;
00082     }
00083 };
```

```
00090 class MIEM : public University
00091 {
00092     int sum = 0;
00093     int div = 11;
00094     std::string sex;
00095     int year;
00096     int month;
00097     int day;
00098     std::string ticket = "";
00099 public:
00100     MIEM(std::string sex,int year, int month, int day)
00101     {
00102         this->sex = sex;
00103         this->year = year;
00104         this->month = month;
00105         this->day = day;
00106     }
00107     std::string generator() override
00108     {
00109
00110         if (sex == "man")
00111         {
00112             ticket+=std::to_string(8);
00113             sum+=8;
00114         }
00115         if (sex == "woman")
00116         {
00117             ticket+=std::to_string(4);
00118             sum+=4;
00119         }
00120         int date = 10000*year + 100*month + day;
00121         ticket+=std::to string(date);
00122         for (int i = 1; i < ticket.size();i++) sum+=(ticket[i] - '0') * (i+1);
00123         std:: mt19937 seq(date + time(0));
00124         std:: uniform_int_distribution<> gap(10000,99999);
00125         std::string rand number = std::to string(gap(seq));
00126         for (int i = 0; i < rand number.size();i++)
sum+=(rand number[i]-'0')*(10+i);
00127         while (sum % div == 4)
00128         {
00129             sum = sum - rand_number[2]*(10+2);
00130             int new d = ((rand number[2] - '0')+1)%10 + '0';
00131             rand number =
{rand number[0],rand number[1],(char)new d,rand number[3],rand number[4]};
00132             sum = sum + new_d*(10+2);
00133         }
00134         ticket+=rand_number;
00135         int ost = sum / div;
00136         for (int i = 0; i < 10;i++)
00137         {
00138             if ((ost + (15*i)%11)%11 == 0)
00139             {
00140                 ost = 0;
00141                 ticket+=std::to string(i);
00142                 break;
00143             }
00144         }
00145         return ticket;
00146     }
00147 };
00148 std::string u c;
00149 int y_c;
00150 int m_c;
00151 int d_c;
00152 std::string s c;
00153
00158 */
00159 class GenOfTicket {
00160     public:
00161     GenOfTicket(){};
00162     University* generate(std::string,std::string,int,int,int);
00163 };
00164 University* GenOfTicket::generate(std::string name,std::string sex,int year,int
month,int day)
00165 {
00166     if (name == "MIEM")
00167     {
```

```
00168         MIEM* univer = new MIEM(sex,year,month,day);
00169         return univer;
00170     }
00171     else
00172     {
00173         MGTU* univer = new MGTU(sex,year,month,day);
00174         return univer;
00175     }
00176 }
00177 std::string gen ticket()
00178 {
00179     GenOfTicket tick_gener;
00180     std::string ticket;
00181     if (s c == "man") ticket =
tick gener.generate(u c,"man",y c,m c,d c)->generator();
00182     else ticket = tick gener.generate(u c,"woman",y c,m c,d c)->generator();
00183     return ticket;
00184 }
00188 void FromMenu(){
00189     std::cout << "To output from a file, enter: ./prog --fromfile <name.txt>" <<
std::endl;
00190 }
00194 void ToMenu(){
00195     std::cout << "To enter the file, enter: ./prog --tofile <name.txt>" << std::endl;
00196 }
00200 void FromToMenu(){
00201     std::cout << "For reading from one file and out to another file enter: ./nameprog
--fromfile --tofile <name1.txt> <name2.txt>" << std::endl;
00202 }
00206 void Menu(){
00207     std::cout << "Enter as follows:" << std::endl;
00208     FromMenu();
00209     ToMenu();
00210     FromToMenu();
00211 }
00216 void CheckDate(int year,int month,int day)
00217 {
00218     int arr[] = {31,29,31,30,31,30,31,31,30,31,30,31};
00219     if (year%4 != 0) arr[1] = 28;
00220     if (!(day <= arr[month-1]) )
00221     {
00222         cerr << "Enter correct date" << endl;
00223         exit(1);
00224     }
00225 }
00231 void ToFile(char *namefile)
00232 {
00233     std::ofstream file(namefile);
00234     std::string name un;
00235     std::string sex;
00236     std::string year_s;
00237     std::string month_s;
00238     std::string day s;
00239     cout << "Enter name university" << endl;
00240     cin >> name un;
00241     if (!(name_un == "MIEM" || name_un == "MGTU"))
00242     {
00243         cerr << "Enter MGTU or MIEM university" << endl;
00244         exit(1);
00245     }
00246     cout << "Enter sex" << endl;
00247     cin >> sex;
00248     if (!(sex == "man" || name_un == "woman") )
00249     {
00250         cerr << "Enter <man> of <woman>" << endl;
00251         exit(1);
00252     }
00253     cout << "Enter year of birthday " << endl;
00254     cin >> year_s;
00255     int year = stoi(year_s);
00256     if (year < 1900 || year > 2004)
00257     {
00258         cerr << "You need enter year in range 1900 to 2004" << endl;
00259         exit(1);
00260     }
00261     cout << "Enter month of birthday in number " << endl;
00262     cin >> month_s;
```

```
00263
00264     int month = stoi(month s);
00265
00266     if (month < 1 || month > 12)
00267     {
00268         cerr << "You need enter month in range 1 to 12" << endl;
00269         exit(1);
00270     }
00271     cout << "Enter day of birthday " << endl;
00272     cin >> day s;
00273     int day = stoi(day_s);
00274     if (day < 1 && day > 31)
00275     {
00276         cerr << "You need enter month in range 1 to 12" << endl;
00277         exit(1);
00278     }
00279     CheckDate(year,month,day);
00280     if (name_un == "MIEM")
00281     {
00282         MIEM obj(sex,year,month,day);
00283         file << obj.generator() << endl;
00284     }
00285     else
00286     {
00287         MGTU ob(sex,year,month,day);
00288         file << ob.generator() << endl;
00289     }
00290     file.close();
00291 }
00297 void FromFile(char *namefile)
00298 {
00299     std::ifstream file(namefile);
00300     if (!file.is open()){ cerr << "No found file " << endl; exit(1);}
00301     std::string name_un= "";
00302     std::string sex = "";
00303     std::string year s = "";
00304     std::string month s = "";
00305     std::string day s = "";
00306     file >> name_un >> sex >> year_s >> month_s >> day_s;
00307     file.close();
00308     if (!(name_un == "MIEM" || name_un == "MGTU"))
00309     {
00310        cerr << "Enter MGTU or MIEM university" << endl;
00311        exit(1);
00312     }
00313
00314
00315     if (!(sex == "man" || name un == "woman") )
00316     {
00317         cerr << "Enter <man> of <woman>" << endl;
00318         exit(1);
00319     }
00320
00321
00322     int year = stoi(year s);
00323     if (year < 1900 || year > 2004)
00324     {
00325         cerr << "You need enter year in range 1900 to 2004" << endl;
00326         exit(1);
00327     }
00328
00329
00330     int month = stoi(month_s);
00331     if (month < 1 || month > 12)
00332     {
00333         cerr << "You need enter month in range 1 to 12" << endl;
00334         exit(1);
00335     }
00336
00337     int day = stoi(day_s);
00338     if (day < 1 && day > 31)
00339     {
00340         cerr << "You need enter month in range 1 to 12" << endl;
00341         exit(1);
00342     }
00343     CheckDate(year,month,day);
00344     if (name_un == "MIEM")
```

```
00345     {
00346         MIEM obj(sex,year,month,day);
00347         cout << obj.generator() << endl;
00348     }
00349     else
00350     {
00351         MGTU ob(sex,year,month,day);
00352         cout << ob.generator() << endl;
00353     }
00354 }
00360 void FromToFile(char* namefile1,char* namefile2)
00361 {
00362     std::ifstream file1(namefile1);
00363     std::ofstream file2(namefile2);
00364     if (!file1.is open()){ cerr << "No found file " << endl; exit(1);}
00365     std::string name un= "";
00366     std::string sex = "";
00367     std::string year_s = "";
00368     std::string month s = "";
00369     std::string day s = "";
00370     file1 >> name un >> sex >> year s >> month s >> day s;
00371     file1.close();
00372     if (!(name_un == "MIEM" || name_un == "MGTU"))
00373     {
00374        cerr << "Enter MGTU or MIEM university" << endl;
00375        exit(1);
00376     }
00377
00378
00379     if (!(sex == "man" || name un == "woman") )
00380     {
00381         cerr << "Enter <man> of <woman>" << endl;
00382         exit(1);
00383     }
00384
00385
00386     int year = stoi(year s);
00387     if (year < 1900 || year > 2004)
00388     {
00389         cerr << "You need enter year in range 1900 to 2004" << endl;
00390         exit(1);
00391     }
00392
00393
00394     int month = stoi(month_s);
00395     if (month < 1 || month > 12)
00396     {
00397         cerr << "You need enter month in range 1 to 12" << endl;
00398         exit(1);
00399     }
00400
00401     int day = stoi(day_s);
00402     if (day < 1 && day > 31)
00403     {
00404         cerr << "You need enter month in range 1 to 12" << endl;
00405         exit(1);
00406     }
00407     CheckDate(year,month,day);
00408     if (name un == "MIEM")
00409     {
00410         MIEM obj(sex,year,month,day);
00411         file2 << obj.generator() << endl;
00412     }
00413     else
00414     {
00415         MGTU ob(sex,year,month,day);
00416         file2 << ob.generator() << endl;
00417     }
00418 }
00419
```

Файл class.h

Функции

- void **FromMenu** ()
- void **ToMenu** ()
- void **FromToMenu** ()
- void **Menu** ()
- void **FromFile** (char *namefile)
- void **FromToFile** (char *in_file, char *to_file)
- void **ToFile** (char *namefile)

Функции

void FromFile (char * *namefile*)

Функция, принимающая название файла, дальше считывающая данные из файла, введенные через пробел, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в консоль учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **297**

void FromMenu ()

Функция для вывода меню-подсказки при использовании –fromfile

См. определение в файле **class.cpp** строка **188**

void FromToFile (char * *namefile1*, char * *namefile2*)

Функция, принимающая название файла, дальше считывающая данные из файла, введенные через пробел, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в файл учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **360**

void FromToMenu ()

Функция для ввода меню-подсказки при использовании –fromfile и –tofile одновременно

См. определение в файле **class.cpp** строка **200**

void Menu ()

Функция для ввода общего меню-подсказки, содержащего выше упомянутые меню

См. определение в файле **class.cpp** строка **206**

void ToFile (char * *namefile*)

Функция, принимающая название файла, дальше требующая данные у пользователя с консоли, в зависимости от университета вызывающая метод определенного класса и выводящая ответ в файл учитываются проверки на месяц(0<n<13), год(1899<y<2005), день(0<d<31) и пол

См. определение в файле **class.cpp** строка **231**

void ToMenu ()

Функция для вывода меню-подсказки при использовании –tofile

См. определение в файле **class.cpp** строка **194**

class.h

См. документацию.
```
00001 #ifndef CLASSES
00002 #define CLASSES
00003 void FromMenu();
00004 void ToMenu();
00005 void FromToMenu();
00006 void Menu();
00007 void FromFile(char* namefile);
00008 void FromToFile(char* in file, char* to file);
00009 void ToFile(char* namefile);
00010 #endif
```

Файл main.cpp

```
#include <iostream>
#include "class.h"
#include <cstring>
#include <string>
```

Функции

- int **main** (int argc, char *argv[])
  Функция main с параметром argc - кол-во аргументов в консоли, argv - массив аргументов

Переменные

- const char * **flags** [2] = {"--fromfile","--tofile"}
  Массив, содержащий строки флагов

Функции

int main (int *argc*, char * *argv*[])

Функция main с параметром argc - кол-во аргументов в консоли, argv - массив аргументов

Если был введен 1 флаг, то пользователь получает меню с подсказкой

Если было введено 2 флага, первый - программа, а второй не –fromfile и –tofile, то пользователь получит ошибку Если 2 флагом были введены выше упомянутые, то вывести меню для корректного ввода каждого флага

Если флагов 3,второй –fromfile или –tofile, а следом название файла, то вызывается соответствующая флагу функция Если пользователь ввел что-то другое, он получает ошибку и меню с подсказкой

Если флагов 5 и было введено след. образом (–fromfile <input_file> –tofile <output_file>, то выполняется соотв. флаг А иначе выводится ошибка и меню-подсказка

Если было введено иное кол-во флагов, то пользователь получает ошибку и меню-подсказку

См. определение в файле **main.cpp** строка **15**

Переменные

const char* flags[2] = {"--fromfile","--tofile"}

Массив, содержащий строки флагов

См. определение в файле **main.cpp** строка **11**

main.cpp

См. документацию.
```
00001 #include <iostream>
00002 #include "class.h"
00003 #include <cstring>
00004 #include <string>
00005 using std::cout;
00006 using std::cerr;
00007 using std::endl;
00011 const char* flags[2] = {"--fromfile","--tofile"};
00015 int main(int argc, char* argv[])
00016 {
00020     if (argc == 1) Menu();
00025     else if (argc == 2)
00026     {
00027         if (!strcmp(argv[1],flags[0])) FromMenu();
00028
00029         else if (!strcmp(argv[1],flags[1])) ToMenu();
00030         else
00031         {
00032             cout << "Wrong flags" << endl;
00033             Menu();
00034         }
00035     }
00040     else if (argc == 3)
00041     {
00042         char* namefile = argv[2];
00043         if (!strcmp(argv[1],flags[1]))
00044         {
00045             ToFile(namefile);
00046         }
00047         else if (!strcmp(argv[1],flags[0]))
00048         {
00049             FromFile(namefile);
00050         }
00051         else
00052         {
00053             cerr << "Wrong flags" << endl;
00054             Menu();
00055         }
00056     }
00061     else if (argc == 5)
00062     {
00063         char *in file = argv[2];
00064         char *to file = argv[4];
00065         if ((!strcmp(argv[1],flags[0])) && (!strcmp(argv[3], flags[1])))
FromToFile(in_file,to_file);
00066         else
00067         {
00068             cerr << "Entered Error" << endl;
00069             FromToMenu();
00070         }
00071     }
00072     else
00076     {
00077         cerr << "You entered wrong count of flags" << endl;
00078         Menu();
00079     }
00080
```