Правительство Российской Федерации

Федеральное государственное автономное
образовательное учреждение высшего профессионального
образования
Национальный исследовательский университет
«Высшая школа экономики»

Московский институт электроники и математики Национального
исследовательского университета «Высшая школа экономики»

ОТЧЕТ
По лабораторным работам № 6-7
по дисциплине “Языки программирования”

Выполнил:
Студент группы СКБ221
Ширинкин Артём Владимирович

Москва, 2022

Постановки задачи:

Требуется сгенерировать студенческий билет вида SYMDNC, S - цифра, отвечающая за пол; Y - год рождения; M - месяц рождения; D - день рождения; N - псевдослучайно число, уникальное для каждой даты рождения; C - число, подобранное так, чтобы сумма цифр студенческого билета умноженных на свой номер слева направо делилась на какое-то k, разное для всех вузов.

Нужно реализовать виртуальный класс-интерфейс для генераторов номеров студбилетов, от которого необходимо отнаследоваться два раза: МИЭМовским и ЭмГэТУУшным классом-генератором.
Код программы должен содержать ввод с консоли флага --tofile и --fromfile. Первый говорит нам, что ввод будет задан в заданный пользователем в командной строке файл (в случае отсутствия файла создать его), а второй говорит о считывании из файла заданного пользователем. Предусмотреть возможность исполнения обоих флагов. Учесть возможность некорректного ввода и обработать исключения.

Также необходимо реализовать класс-генератор общего вида, используя паттерн Шаблонный Метод (необходимо изучить самостоятельно, вот пример кода с комментариями с данным паттерном).

Создать Makefile с возможностью сборки, clean, distclean.

Алгоритм выполнения

1. Функция main(int argc, char** argv) обрабатывает флаги и определяет откуда производится чтение данных и куда должна произвестись запись, вызывает функцию generate_ticket().
2. Функция generate_ticket() при помощи шаблонного класса генератора ticketGenerator создает билет и возвращает его.
3. Шаблонный класс ticketGenerator имеет метод ticketGenerator(name), которая создает объект класса MIEMTicket или MGTUUTicket в зависимости от имени вуза, которое ему передали.
4. Классы MIEMTicket и MGTUUTicket наследуются от абстрактного класса-интерфейса studentTicket и имеют функцию generate(char s, int y, int m, int d). Данная функция генерирует билет по данным в условиях задачи правилам.
5. Также в main.cpp реализованы функции
    a. check_date(int y, int m, int d), check_university_name(std::string name), check_sex(std::string sex), проверяющие введенные данные
    b. to_upper(char c) и to_lower(char c) приводящие строки в удобный для обратки вид
    c. Перегруженная функция read()
        i. read() - считывает данные со стандартного потока ввода
        ii. read(std::string filename) - считывает данные из файла
    d. Перегруженная функция print()
        i. print(std::string ticket) - записывает билет в стандартный потом вывод
        ii. print(std::string ticket, std::string filename) - записывает билет в файл

Код проекта

main.cpp

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

namespace sex{
    char man = 'm';
    char woman = 'w';
};

std::string name;
std::string _sex;
int y;
int m;
int d;


int check_date(int y, int m, int d){
    int days_in_month[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
31};
    if (!(y % 4) && (y % 100)){
        days_in_month[1] = 29;
    }
    if (y > 2022 || y < 1900){
        std::cerr << "Wrong year";
        return 0;
    }
    else if (m > 12 || m < 1){
        std::cerr << "Wrong month";
        return 0;
    }
    else if (d < 0 || d > days_in_month[m - 1]){
        std::cerr << "Wrong date";
        return 0;
    }
```

```
    return 1;
}

int check_university_name(std::string name){
    if (name == "MIEM" || name == "MGTUU"){
        return 1;
    }
    std::cerr << "Wrong university name";
    return 0;
}

int check_sex(std::string sex){
    if (sex == "man" || sex == "woman"){
        return 1;
    }
    std::cerr << "Wrong sex";
    return 0;
}

char to_upper(char c) { return std::toupper(c); }
char to_lower(char c) { return std::tolower(c); }


int read() {
    std::cin >> name;
    std::transform(name.begin(), name.end(), name.begin(), to_upper);
    std::cin >> _sex;
    std::transform(_sex.begin(), _sex.end(), _sex.begin(), to_lower);
    std::cin >> y >>  m >> d;

    return check_date(y, m, d) && check_sex(_sex) &&
check_university_name(name);
}

int read(std::string filename) {
    std::ifstream file (filename);
    if (!file.is_open()) {
```

```
        std::cerr << "Wrong file name";
        return 1;
    }

    std::getline(file, name);
    std::transform(name.begin(), name.end(), name.begin(), to_upper);
    std::getline(file, _sex);
    std::transform(_sex.begin(), _sex.end(), _sex.begin(), to_lower);
    file >> y >> m >> d;

    file.close();

    return check_date(y, m, d) && check_sex(_sex) &&
check_university_name(name);
}

void print(std::string ticket) { std::cout << "Ticket: " << ticket << std::endl;
}

void print(std::string ticket, std::string filename) {
    std::ofstream file (filename);
    file << "Ticket: " << ticket << '\n';
    file.close();
}

std::string generate_ticket() {
    ticketGenerator ticket_generator;
    std::string ticket;
    if (_sex == "man")
        ticket = ticket_generator.generator(name)->generate(sex::man, y,
m, d);
    else
        ticket = ticket_generator.generator(name)->generate(sex::woman,
y, m, d);
    return ticket;
}
```

```
int main(int argc, char ** argv){
    std::string ticket;
    switch (argc){
        case 1:
            if (!read()){
                exit(1);
            }
            ticket = generate_ticket();
            print(ticket);
            break;
        case 2:
            if (!strcmp(argv[1], "--fromfile")) {
                if (!read(argv[2])) exit(1);
                ticket = generate_ticket();
                print(ticket);
            }
            else if (!strcmp(argv[1], "--tofile")) {
                if (!read()) exit(1);
                ticket = generate_ticket();
                print(ticket, argv[2]);
            }
            else {
                std::cerr << "Wrong flags";
                exit(1);
            }
            break;
        case 5:
            if (!strcmp(argv[1], "--tofile") && !strcmp(argv[3], "--fromfile")) {
                if (!read(argv[4])) exit(1);
                ticket = generate_ticket();
                print(ticket, argv[2]);
            }
            else if  (!strcmp(argv[3], "--tofile") && !strcmp(argv[1],
"--fromfile")) {
                if (!read(argv[2])) exit(1);
                ticket = generate_ticket();
```

```
            print(ticket, argv[4]);
        }
        else {
            std::cerr << "Wrong flags";
            exit(1);
        }
        break;
    default:
        std::cerr << "Wrong flags";
        exit(1);
        break;
  }
  return 1;
}
```

classes.cpp

```
#include "classes.h"

std::string MIEMTicket::generate(char s, int y, int m, int d){
  std::string ticket = "";
  int summa = 0;

  if (s == 'm'){
    ticket += std::to_string(man);
    summa += man;
  }
  else{
    ticket += std::to_string(woman);
    summa += woman;
  }

  ticket += std::to_string(10000 * y + 100 * m + d);
  for (int i = 1; i < ticket.size(); ++i){
    summa += (ticket[i] - '0') * (i + 1);
  }

  std::mt19937 gen(y + m + d);
```

```
    std::uniform_int_distribution<> distr(10000, 99999);
    std::string id = std::to_string(distr(gen));

    for (int i = 0; i < id.size(); ++i){
        summa += (id[i] - '0') * (i + 10);
    }

    if (summa % 11 == 4){
        summa -= id[0] * 10;
        char id0 = (char)(((id[0] - '0' + 1) % 10) + '0');
        ticket += id0;
        ticket += id.substr(1);
        summa += (id0 - '0') * 10;
    }

    int remainder = summa % 11;
    for (int i = 0; i < 10; ++i){
        if (!((i * 15 + remainder) % 11)){
            ticket += std::to_string(i);
            break;
        }
    }

    return ticket;
}

std::string MGTUUTicket::generate(char s, int y, int m, int d){
    std::string ticket = "";
    int summa = 0;

    if (s == 'm'){
        ticket += std::to_string(man);
        summa += man;
    }
    else{
        ticket += std::to_string(woman);
        summa += woman;
```

```
    }

    ticket += std::to_string(10000 * y + 100 * m + d);
    for (int i = 1; i < ticket.size(); ++i){
        summa += (ticket[i] - '0') * (i + 1);
    }

    std::mt19937 gen(y + m + d);
    std::uniform_int_distribution<> distr(1000, 9999);
    std::string id = std::to_string(distr(gen));

    for (int i = 0; i < id.size(); ++i){
        summa += (id[i] - '0') * (i + 10);
    }

    if ((summa % 10) % 2){
        summa -= id[1] * 11;
        char id1 = (char)(((id[1] - '0' + 1) % 10) + '0');
        ticket += id[0];
        ticket += id1 + id.substr(2);
        summa += (id1 - '0') * 11;
    }

    int remainder = summa % 10;
    for (int i = 0; i < 10; ++i){
        if (!((summa + i * 14) % 10)){
            ticket += std::to_string(i);
            break;
        }
    }

    return ticket;
}

studentTicket *ticketGenerator::generator(std::string universityName){
    if (universityName == "MIEM"){
        MIEMTicket *ticket = new MIEMTicket;
```

```
        return ticket;
    }
    else if (universityName == "MGTUU"){
        MGTUUTicket *ticket = new MGTUUTicket;
        return ticket;
    }
    else{
        std::cerr << "Wrong university name" << std::endl;
        exit(1);
    }
}
```

classes.h

```
#ifndef guard
#define guard

#include <string>
#include <ctime>
#include <iomanip>
#include <random>
#include <iostream>

class studentTicket{
    public:
        virtual std::string generate(char, int, int, int) = 0;
};

class MIEMTicket: public studentTicket{
        int man = 8;
        int woman = 4;

    public:
        MIEMTicket(){}
        std::string generate(char, int, int, int) override;
};

class MGTUUTicket: public studentTicket{
```

```
        int man = 2;
        int woman = 1;

    public:
        MGTUUTicket(){}
        std::string generate(char, int, int, int) override;
};

class ticketGenerator{
    public:
        ticketGenerator(){}
        studentTicket *generator(std::string);
};

#endif
```