**Московский авиационный институт (национальный исследовательский университет)**

Институт №3. Факультет «Системы управления, информатика и электроэнергетика». Кафедра №304

**Пояснительная записка к курсовой работе**

по дисциплине «Программирование на языке высокого уровня»

«Разработка средствами С++ информационной системы поддержки работы бухгалтерии предприятия»

Выполнил студент группы М30-207Б-18

Гордеев Н. М. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Принял к.т.н., доцент

**Офицеров В.П.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва 2019**

**Задание на курсовую работу по дисциплине**

«Программирование на языке высокого уровня»

студенту группы М30-207Б-18

Гордееву Никите Максимовичу

Тема: «Разработка средствами С++ информационной системы поддержки работы бухгалтерии предприятия»

Задание:

1. Сформулировать функциональные и не функциональные требования к информационной файловой системе.
2. Спроектировать структуры данных для заданной предметной области.
3. Реализовать информационную систему средствами С++.

Решаемые задачи:

• учет труда и зарплаты работников;

• табель учета рабочего времени составляется по подразделениям и включает ФИО работника, должность, количество отработанных часов и количество нерабочих дней за месяц;

• формирование ведомостей на выплату зарплаты.

Срок начала: 2 сентября 2019 г.

Срок окончания: 15 декабря 2019 г.

Руководитель:

доцент, к.т.н. Офицеров Владимир Петрович

(Ф.И.О)

Исполнитель:

студент группы M3О-207Б Гордеев Никита Максимович

(Ф.И.О)

Содержание

[1. Определение функциональных требований к системе 4](#__RefHeading___Toc2803_83763553)

[1.1 Моделирование среды функционирования бухгалтерии предприятия 4](#__RefHeading___Toc2788_83763553)

[1.2 Моделирование функций бухгалтерии предприятия 5](#__RefHeading___Toc2790_83763553)

[1.3 Формирование функциональных требований 6](#__RefHeading___Toc2792_83763553)

[2 Техническое задание 7](#__RefHeading___Toc2794_83763553)

[3. Проектирование информационной системы поддержки работы бухгалтерии предприятия 8](#__RefHeading___Toc2796_83763553)

[3.1 Модель среды функционирования информационной системы 8](#__RefHeading___Toc2806_83763553)

[3.2 Моделирование работы информационной системы 9](#__RefHeading___Toc2798_83763553)

[3.3 Реализация системы информационной поддержки работы бухгалтерии предприятия 12](#__RefHeading___Toc2808_83763553)

[3.3.1 Фрагменты текста программы реализующие основные функциональные требования 12](#__RefHeading___Toc2815_83763553)

[3.3.2 Примеры работы системы 20](#__RefHeading___Toc2810_83763553)

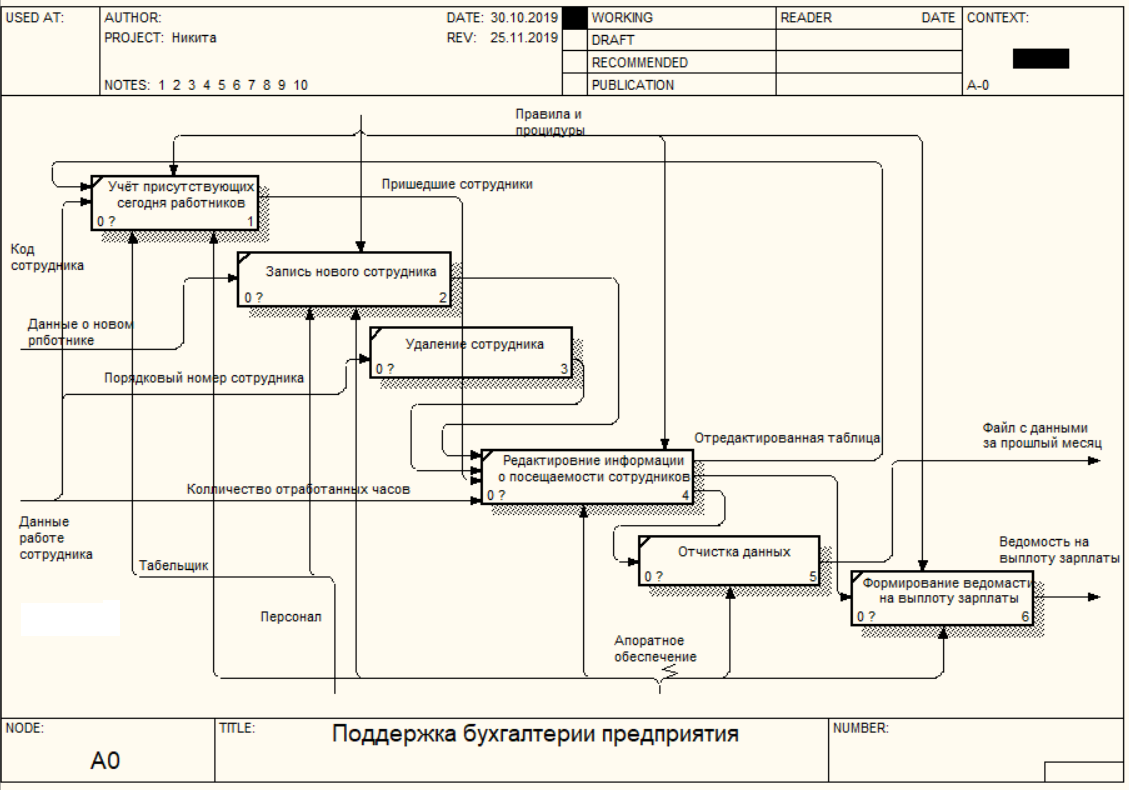
[4. Вывод 26](#__RefHeading___Toc2812_83763553)

[Приложение А. Исходные коды информационной системы поддержки бухгалтерии предприятия 27](#__RefHeading___Toc2817_83763553)

# 1. Определение функциональных требований к системе

## 1.1 Моделирование среды функционирования бухгалтерии предприятия

## 1.2 Моделирование функций бухгалтерии предприятия



## 1.3 Формирование функциональных требований

1) Учёт присутствующих сегодня работников

Цель — учет пришедших сегодня работников.

Принимает на вход:

a) Целое, положительное число, меньше 32, означающее сегодняшнее число.

б) Целое положительное число, обозначающее персональный код сотрудника.

Передаёт информацию о присутствующих работниках.

2) Запись нового сотрудника.

Цель — добавить работника в архив.

Принимает на вход

а) 3 набора символов, разделённых пробелом и не включающих пробел, табуляцию и знак переноса строки, обозначающих ФИО сотрудника.

б) Целое число от 1 до 12, обозначающее код должности сотрудника.

в) Целое положительное число, обозначающее персональный код сотрудника.

Передает исходный архив с дописанными в конец файла данными о новом сотруднике

3) Удаление сотрудника.

Цель — удалить работника из архива.

Принимает на вход

а)Целое, положительное число, обозначающее порядковый номер сотрудника в списке

Передает исходный архив с отсутствующими данными о новом сотруднике

4) Редактирование информации о посещаемости сотрудников

Цель — записать актуальные данные о работниках.

Принимает на вход

а) Целое число, обозначающее отработанные часы конкретного работника.

б) Не отсортированный архив с сотрудниками.

Передает готовый к использованию архив данных.

5) Очистка данных

Цель — удалить устаревшую информацию о сотрудниках, сохранив старый архив.

Принимает на вход старый архив

Возвращает отчищенный архив и копию старого.

6) Формирование ведомости на выплату зарплаты.

Принимает на вход архив данных о работниках.

Возвращает ведомость на выплату зарплаты.

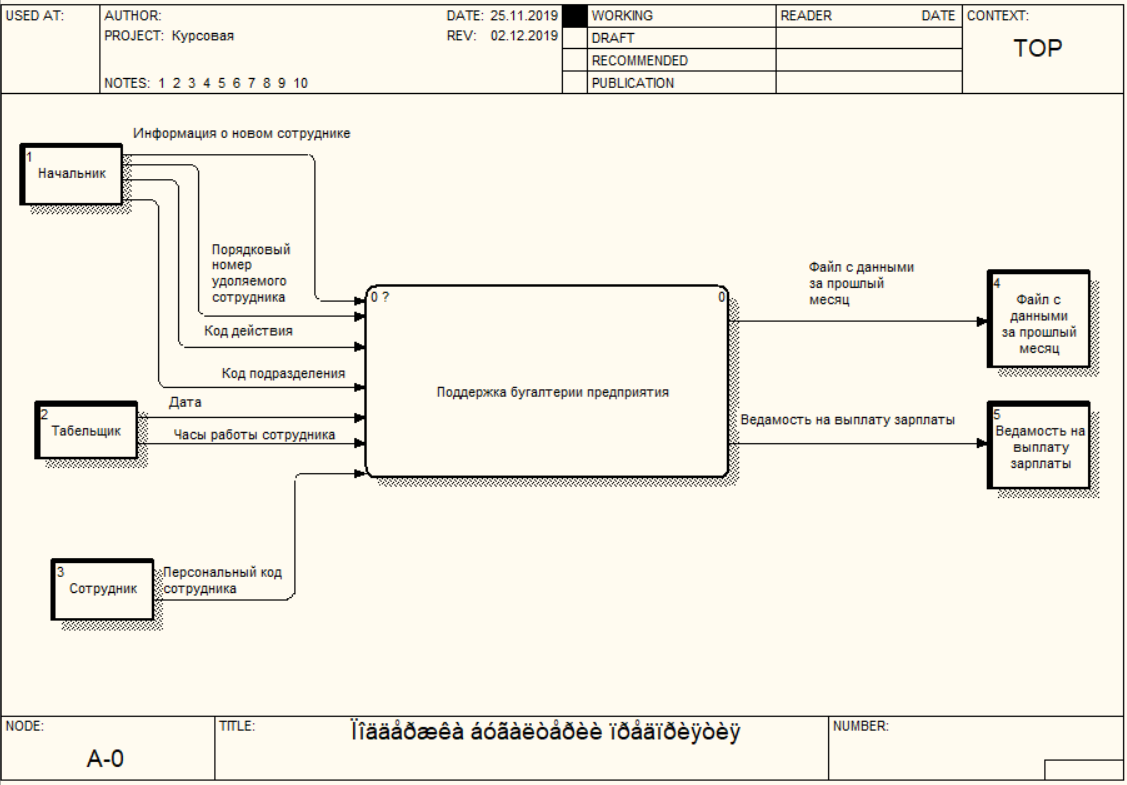
# 2 Техническое задание

Разработать информационную систему поддержки деятельности бухгалтерии предприятия с помощью средств языка “C++” реализующий следующие функциональные и не функциональные требования:

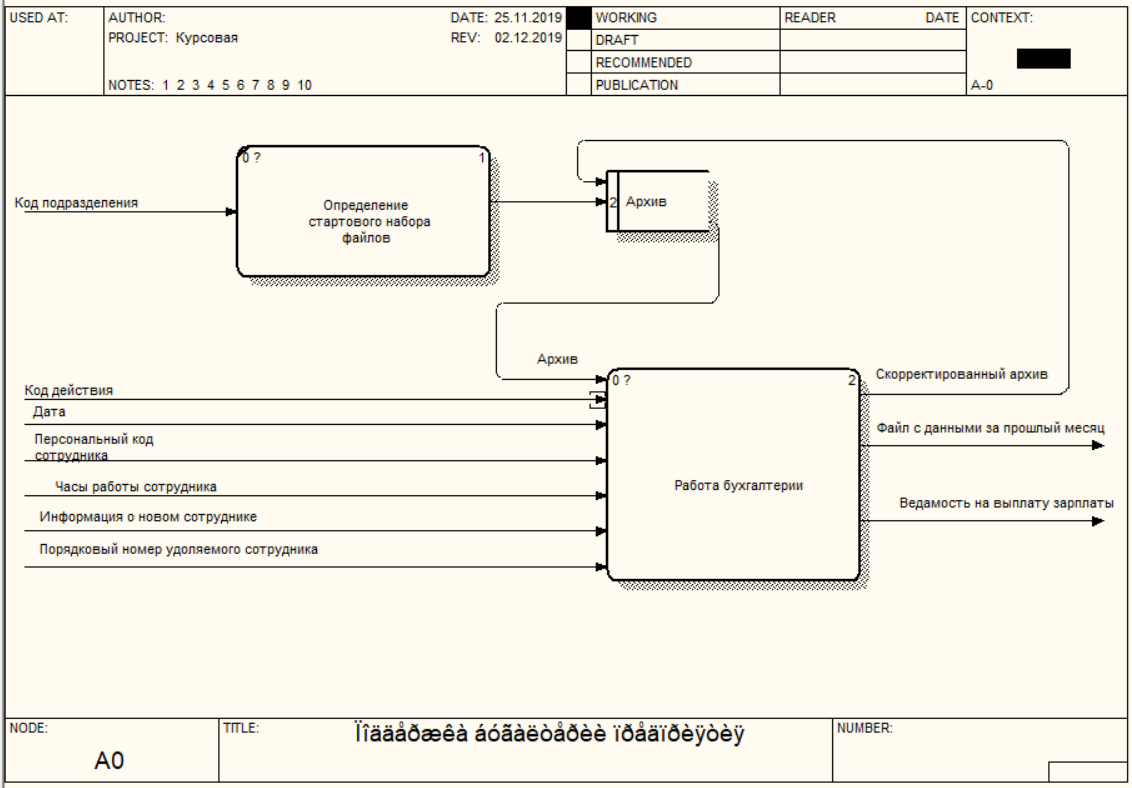
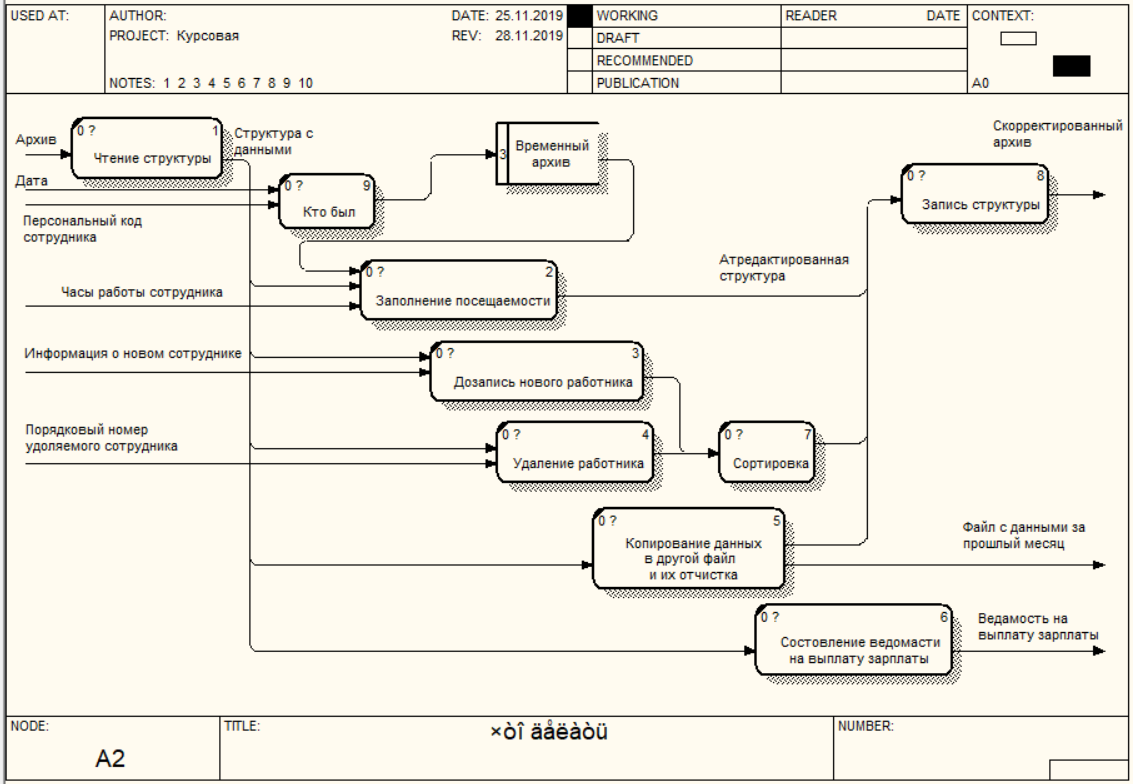
|  |  |
| --- | --- |
| Функциональные | Не функциональные |
| Реализовать возможность выбора подразделения при запуске программы | По введённому в консоль числу запустить стартовый набор файлов |
| Учет присутствующих сегодня работников | Реализовать запись в файл даты и персональных кодов сотрудников |
| Запись нового работника | Увеличение размера массива структур, считывание введенных в консоль данных о работнике, добавление нового работника,  отсортировать данные. |
| Удаление из списка работника по его порядковому номеру | Поиск по массиву структур студента с введенными номером и удаление из массива структур |
| Редактирование информации о  посещаемости сотрудников | Запросить ввод времени работы присутствующих сотрудников |
| Очистка данных | Скопировать имеющиеся данные о сотрудниках в другой файл и обнулить массив часов работ сотрудников. |
| Формирование ведомости на выплату зарплаты | Вывести на экран ФИО, порядковый номер, должность, количество отработанных дней, количество отработанных часов в конкретный день, зарплату каждого сотрудника. |

# **3. Проектирование информационной системы поддержки работы бухгалтерии предприятия**

## 3.1 Модель среды функционирования информационной системы

****

## 3.2 Моделирование работы информационной системы



1) Определение стартового набора файлов

Запросить ввести число

Если ввели 1 передать дальше 1 файл

Если ввели 2 передать дальше 2 файл

итд.

2) Что делать

Запросить пароль

если пароль верный

запустить Чтение структуры

Запросить ввод числа

если ввели 1

запустить Кто был

если ввели 2

запустить Дозапись нового работника

если ввели 3

запустить Удаление работника

если ввели 4

запустить Копирование данных в другой файл и их отчистка

если ввели 5

запустить Составление ведомости на выплату зарплаты

3) Чтение структуры

Считать количество элементов структуры в N

Считать N раз структуру

4) Кто был

Запросить ввод даты и записать ее в файл

Пока не введут 0

Запросить ввод кода сотрудника и записать его в файл

Запустить Заполнение посещаемости

5) Заполнение посещаемости

Считать дату

Сравнить записанные коды с существующими и если сотрудник был запомнить это

Вывести имена присутствующих работников и запросить ввести количество отработанных ими часов

Запустить Запись структуры

6) Запись структуры

Записать количество элементов структуры (N)

Записать N элементов структуры

7) Дозапись нового работника

Запросить ввод данных о работнике и записать их в конец файла

Запустить Сортировку

Запустить Запись структуры

8) Сортировка

Отсортировать структуру

9) Удаление работника

Запросить ввод номера удаляемого сотрудника

Записать вместо его имени «яяяяяяяяяяяяяяяяяяяяяяяяяяяя»

Запустить сортировку

N уменьшить на 1

Запустить Запись структуры

10) Копирование данных в другой файл и их очистка

Запустить Запись структуры, передав файл для прощломесячных данных

Обнулить графу часы работу в массиве структур

Запустить Запись структуры

11) Составление ведомости на выплату зарплаты

Нарисовать таблицу с данными, полями в которых будут:

Порядковый номер сотрудника

ФИО сотрудника

Должность сотрудника

Количество отработанных дней

В какой день сколько отработал

Зарплата = все отработанные часы \* зарплата в час

## 3.3 Реализация системы информационной поддержки работы бухгалтерии предприятия

### 3.3.1 Фрагменты текста программы реализующие основные функциональные требования

**1) Учёт присутствующих сегодня работников**

**void Кто\_был(const char FNAME1[]) {**

FILE\* fp = fopen(FNAME1, "wb");

int Код\_сотрудника = -1;

int число = 32;

system("cls"); //отчистка экрана

//ввод числа

while (число < 1 || число > 31)

{

cout << "Введите сегодняшнее число: ";

cin >> число;

//защита от дураков

if (число < 1 || число > 31) {

system("cls");

cout << "Неверный ввод числа\n";

}

else

fwrite(&число, sizeof(int), 1, fp);

}

//пока не введут 0

while (Код\_сотрудника != 0)

{

system("cls"); //отчистка экрана

cout << "Введите свой персональный код: ";

cin >> Код\_сотрудника;

fwrite(&Код\_сотрудника, sizeof(Код\_сотрудника), 1, fp);

}

fclose(fp);

system("cls"); //отчистка экрана

}

**2) Запись нового сотрудника.**

void Запись\_нового\_сотрудника(struct структура\* работник, int N, const char FNAME[]) {

system("cls"); //отчистка экрана

int Код\_сотрудника;

int t = 0;

структура новый\_сотрудник;

cout << "Введите ФИО сотрудника: ";

cin >> новый\_сотрудник.ФИО[0] >> новый\_сотрудник.ФИО[1] >> новый\_сотрудник.ФИО[2];

system("cls"); //отчистка экрана

cout << "Введите код должности сотрудника из представленных ";

//ввод должности сотрудника

while (t < 1 || t > 12)

{

system("cls"); //отчистка экрана

//печать возможных должностей

for (int i = 0; i < 12; i++)

cout << "\n" << Должности[i] << ": " << i + 1;

cout << "\n\nКод должности сотрудника: ";

cin >> t;

//защита от дураков

if (t < 1 || t > 12)

{

cout << "\nНет такой должности\n";

system("PAUSE");

}

}

новый\_сотрудник.должность = t;

system("cls"); //отчистка экрана

//ввод личного кода сотрудника

t = 0;

do {

if (t)

cout << "\nсотрудник с таким кодом уже существует\n";

cout << "Персональный код сотрудника: ";

cin >> новый\_сотрудник.код;

t = 0;

//код уже существует

for (int i = 0; i < N; i++)

{

if (работник[i].код == новый\_сотрудник.код)

t = 1;

}

}while (t);

//создали новый масив структур работников

//переписали в него старый а в конец

//нового сотрудника

структура\* работник2 = new структура[N+1];

for (int i = 0; i < N; i++)

{

работник2[i] = работник[i];

}работник2[N] = новый\_сотрудник;

Сортировка(работник2, N + 1, FNAME);

запись\_структуры(работник2, N + 1, FNAME);

delete[] работник2;

**}**

**3) Удаление сотрудника.**

void удаление(struct структура\* работник, int N, const char FNAME[])

{

int уволеный;

system("cls"); //отчистка экрана

cout << "Введите порядковый номер уволившегося сотрудника: ";

cin >> уволеный;

if (уволеный > N || уволеный < 1)

{

cout << "\nНет такого сотрудника\n"; system("PAUSE");

return;

}

for (int i = 0; i < max\_ФИО/3; i++)

работник[уволеный - 1].ФИО[0][i] = 'я';

Сортировка(работник, N, FNAME);

запись\_структуры(работник, N - 1, FNAME);

}

**4) Редактирование информации о посещаемости сотрудников**

void Заполнение\_посищаемости(struct структура\* работник, int N, const char FNAME[], const char FNAME1[]) {

system("cls"); //отчистка экрана

int Часы\_работы;

int Число;

int N1 = 0; //сколько людей записалось

int j, i;

Кто\_был(FNAME1);

//считаю количество записавшихся

FILE\* in1 = fopen(FNAME1, "rb");

fread(&Число, sizeof(int), 1, in1);//чтение числа

while (fread(&i, sizeof(int), 1, in1) == 1) {

N1++;

}fclose(in1);

//чтение принятых индификаторов присуствия

FILE\* out1 = fopen(FNAME1, "rb");

int\* ЛКод = new int[N1];

fread(&i, sizeof(int), 1, out1);//уход с 1-го элемента

for (i = 0; i < N1; i++)

fread(&ЛКод[i], sizeof(int), 1, out1);

fclose(out1);

//создание и обнуление флагов присуствия

bool\* f = new bool[N];

for (j = 0; j < N; j++)

f[j] = 0;

//сравнение присутствуюших с базай

for (j = 0; j < N; j++)

for (i = 0; i < N1; i++)

if (ЛКод[i] == работник[j].код)

f[j] = 1;

//дополнение данных

for (j = 0; j < N; j++) {

if (f[j]) {

Часы\_работы = -1;//опустить флаг

//ввод часов работы

while (Часы\_работы < 0 || Часы\_работы > 24)

{

for (int i = 0; i < 3; i++)

cout << работник[j].ФИО[i] << " ";

cout << "\nОн сегодня работал: ";

cin >> Часы\_работы;

//защита от дурака

if (Часы\_работы < 0 || Часы\_работы > 24) {

system("cls");

cout << "Он не мог столько работать\n";

}

else

работник[j].часы[Число - 1] = Часы\_работы;

}

}

system("cls"); //отчистко экрана

}

запись\_структуры(работник, N, FNAME);

//запись\_структуры(работник, N, FNAME, FNAME2);

delete[] ЛКод; delete[] f;

**}**

**5) Очистка данных**

**/\* Сохранение списка. \*/**

void запись\_структуры(struct структура\* работник, int N, const char FNAME[])

{

FILE\* fp;

register int i;

if ((fp = fopen(FNAME, "wb")) == NULL) {

printf("Ошибка при открытии файла.\n");

return;

}

fwrite(&N, sizeof(N),1, fp);

for (i = 0; i < N; i++)

if (fwrite(&работник[i],

sizeof(struct структура), 1, fp) != 1)

printf("Ошибка при записи файла.\n");

fclose(fp);

**}**

**void перезапись(struct структура\* работник, int N, const char FNAME[])**

{

system("cls"); //отчистка экрана

char s[256]; //буфер

запись\_структуры(работник, N, FNAME\_копия);

//обнуление часов

for (int j = 0; j < N; j++)

for (int i = 0; i < 31; i++)

работник[j].часы[i] = 0;

запись\_структуры(работник, N, FNAME);

cout << "Перезапись прошла успешно, данные за прошлый месяц хранятся в файле \"Text\_copi\"\n";

system("PAUSE");

**}**

**6) Формирование ведомости на выплату зарплаты.**

//подключение английского языка

void a() {

SetConsoleCP(866);

SetConsoleOutputCP(866);

}

//подключение русского языка

void r() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

}

void шапка(int max, int max\_должность) {

//начало шапки

a();

cout << char(218) << setfill(char(196)) << setw(6);

cout << char(194) << setfill(char(196)) << setw(max + 3);

cout << char(194) << setfill(char(196)) << setw(max\_должность + 3);

cout << char(194) << setfill(char(196)) << setw(13);

for (size\_t m = 0; m < 31; m++)

cout << char(194) << setfill(char(196)) << setw(3);

cout << char(194) << setfill(char(196)) << setw(13);

cout << char(191);

cout << "\n";

//шапка номера

cout << char(179);

r(); cout << " № "; a();

//шапка ФИО

int i = (max + 3) / 2;

cout << char(179);

r(); cout << setfill(char(32)) << setw(i) << " ФИО"; a();

if ((max + 3) % 2 == 1)

i += 1;

cout << setfill(char(32)) << setw(i);

//шапка должности

cout << char(179);

r(); cout << " Должность "; a();

//шапка рабочих дней

cout << char(179);

r(); cout << "Рабочих дней"; a();

//шапки часов работ

for (size\_t m = 0; m < 31; m++)

cout << char(179) << setfill(char(32)) << setw(2) << m + 1;

//шапка зарплаты

cout << char(179);

r(); cout << " Зарплата "; a();

cout << char(179);

cout << "\n";

}

void отчерк() {

cout << char(195) << setfill(char(196)) << setw(6); //номер

cout << char(197) << setfill(char(196)) << setw(max\_ФИО + 3); //ФИО

cout << char(197) << setfill(char(196)) << setw(max\_должность + 3); //должность

cout << char(197) << setfill(char(196)) << setw(13); //рабочие дни

for (size\_t m = 0; m < 31; m++) //часы работ

cout << char(197) << setfill(char(196)) << setw(3);

cout << char(197) << setfill(char(196)) << setw(13) << char(180); //зарплата

cout << "\n";

}

void серкдина(int N, string ФИО, int Должность, int Час[31], int колличество\_рабочих\_дней, int колличество\_отработанных\_часов) {

int Зарплата[12] = { 100,150,200,150,10000,1000,80,90,5000,1200,100,50 };

cout << char(179) << " " << setfill(char(32)) << setw(3) << N << " ";

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(max\_ФИО) << ФИО << " " << right; a();

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(max\_должность) << Должности[Должность - 1] << " " << right; a();

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(10) << колличество\_рабочих\_дней << " " << right; a();

for (size\_t i = 0; i < 31; i++)

{

if (Час[i])

cout << char(179) << setfill(char(32)) << setw(2) << Час[i];

else

cout << char(179) << " H";

}

cout << char(179);

cout << " " << setfill(char(32)) << setw(10) << колличество\_отработанных\_часов \* Зарплата[Должность - 1] << " ";

cout << char(179) << "\n";

}

void конец() {

cout << char(192) << setfill(char(196)) << setw(6); //номер

cout << char(193) << setfill(char(196)) << setw(max\_ФИО + 3); //ФИО

cout << char(193) << setfill(char(196)) << setw(max\_должность + 3); //должность

cout << char(193) << setfill(char(196)) << setw(13); //рабочие дни

for (size\_t m = 0; m < 31; m++) //часы работ

cout << char(193) << setfill(char(196)) << setw(3);

cout << char(193) << setfill(char(196)) << setw(13) << char(217); //зарплата

cout << "\n\n\n"; r();

}

void Состовление\_ведомости(struct структура\* работник, int N, const char FNAME[]) {

int Должность;

int i, j;

string ФИО;

system("cls");

шапка(max\_ФИО, max\_должность); //печать начала таблици

for (i = 0; i < N; i++)

{

for (j = 0; j < 3; j++) {

ФИО.append(работник[i].ФИО[j]); //дописать в конец ФИО

ФИО.push\_back(' ');//дописать пробел

}

//считаю колличество рабочих дней

int колличество\_рабочих\_дней = 0;

for (size\_t m = 0; m < 31; m++)

if (работник[i].часы[m])

колличество\_рабочих\_дней++;

int колличество\_отработанных\_часов = 0;

for (size\_t m = 0; m < 31; m++)

колличество\_отработанных\_часов += работник[i].часы[m];

отчерк(); //рисую отчерк

серкдина(i + 1, ФИО, работник[i].должность, работник[i].часы, колличество\_рабочих\_дней, колличество\_отработанных\_часов);

ФИО.clear(); //отчистка буферов

}

конец();

system("PAUSE");

}

void перезапись(struct структура\* работник, int N, const char FNAME[])

{

system("cls"); //отчистка экрана

char s[256]; //буфер

запись\_структуры(работник, N, FNAME\_копия);

//обнуление часов

for (int j = 0; j < N; j++)

for (int i = 0; i < 31; i++)

работник[j].часы[i] = 0;

запись\_структуры(работник, N, FNAME);

cout << "Перезапись прошла успешно, данные за прошлый месяц хранятся в файле \"Text\_copi\"\n";

system("PAUSE");

**}**

### 3.3.2 Примеры работы системы

Начало:

Введите номер подразделения: 1

Введите пароль: 3

Основные возможности:

1)Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

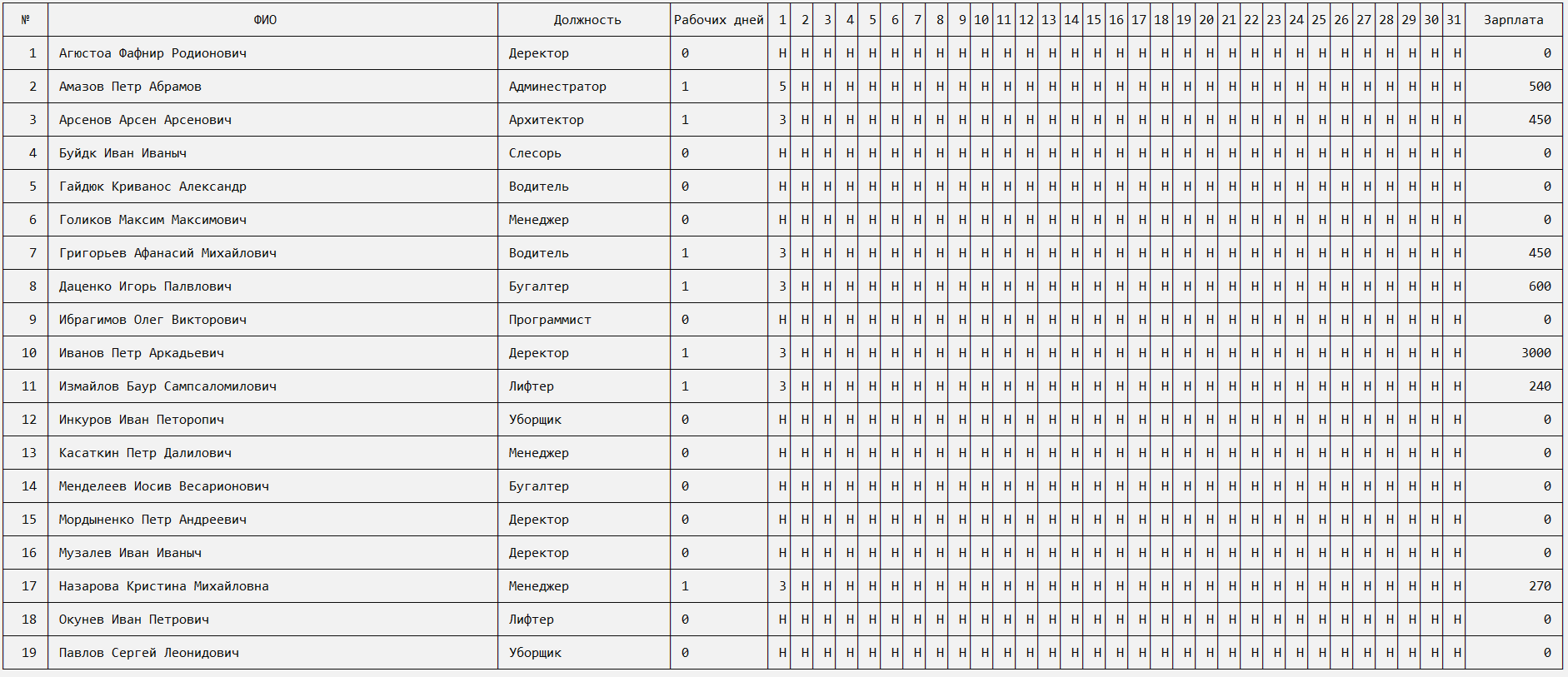
Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 3



2)Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 1

Введите сегодняшнее число: 2

Введите свой персональный код: 1

Введите свой персональный код: 2

Введите свой персональный код: 3

Введите свой персональный код: 0

Амазов Петр Абрамов

Он сегодня работал: 5

Арсенов Арсен Арсенович

Он сегодня работал: -1

Он не мог столько работать

Арсенов Арсен Арсенович

Он сегодня работал: 25

Он не мог столько работать

Арсенов Арсен Арсенович

Он сегодня работал: 7

Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

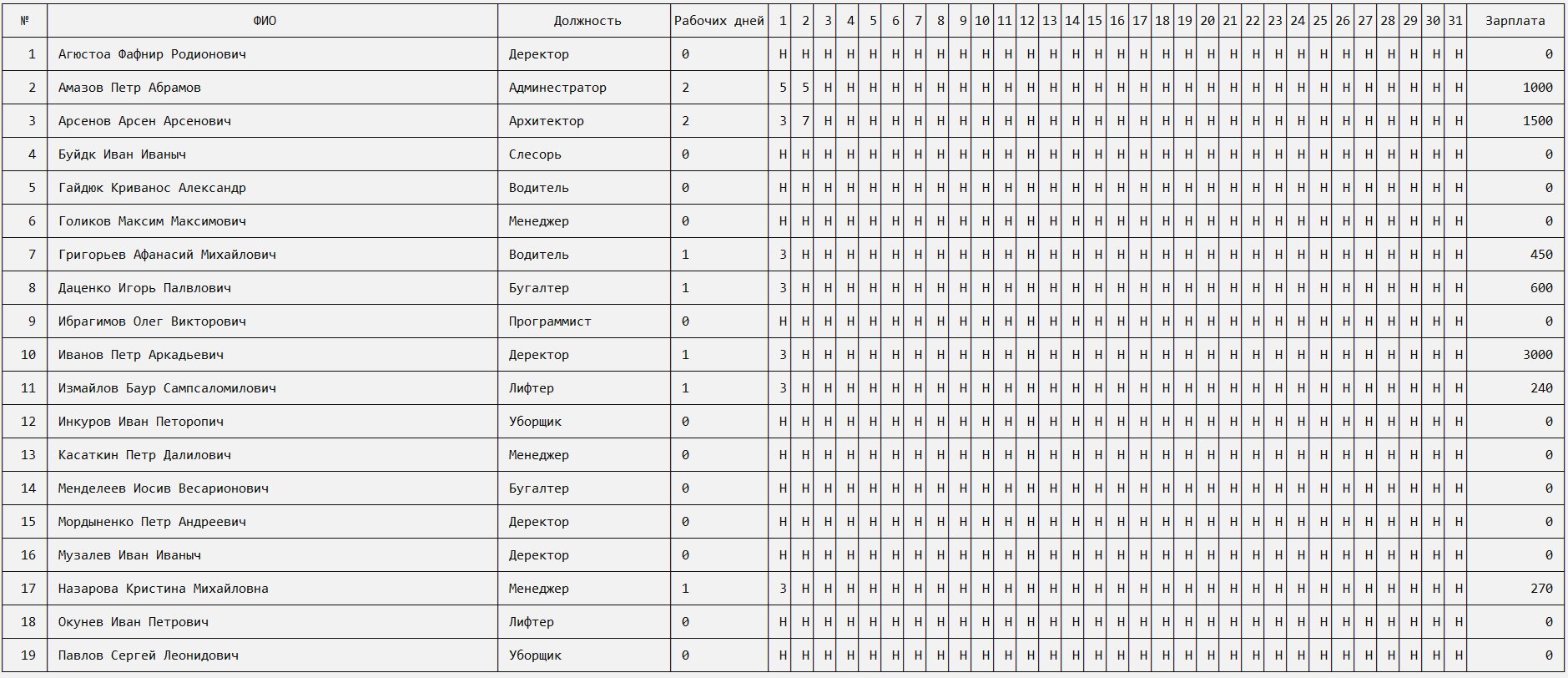
Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 3



3)Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 2

Введите ФИО сотрудника: Измайлов Виктор Федорович

Админестратор: 1

Архитектор: 2

Бугалтер: 3

Водитель: 4

Генеральный деректор: 5

Деректор: 6

Лифтер: 7

Менеджер: 8

Программист: 9

Пилот: 10

Слесорь: 11

Уборщик: 12

Код должности сотрудника: 4

Персональный код сотрудника: 1

сотрудник с таким кодом уже существует

Персональный код сотрудника: 4

сотрудник с таким кодом уже существует

Персональный код сотрудника: 44

Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

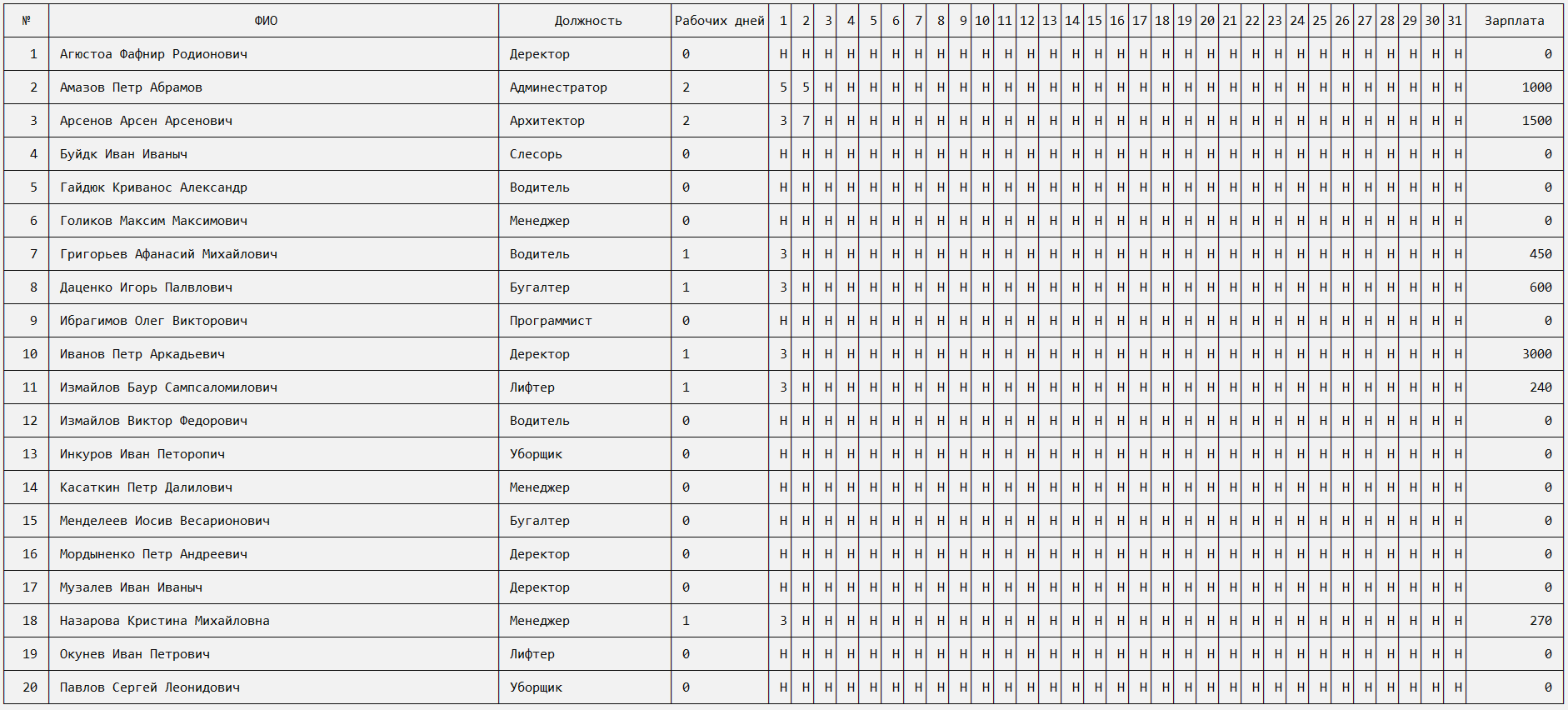
Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 3



4)Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 5

Введите порядковый номер уволившегося сотрудника: 11

Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

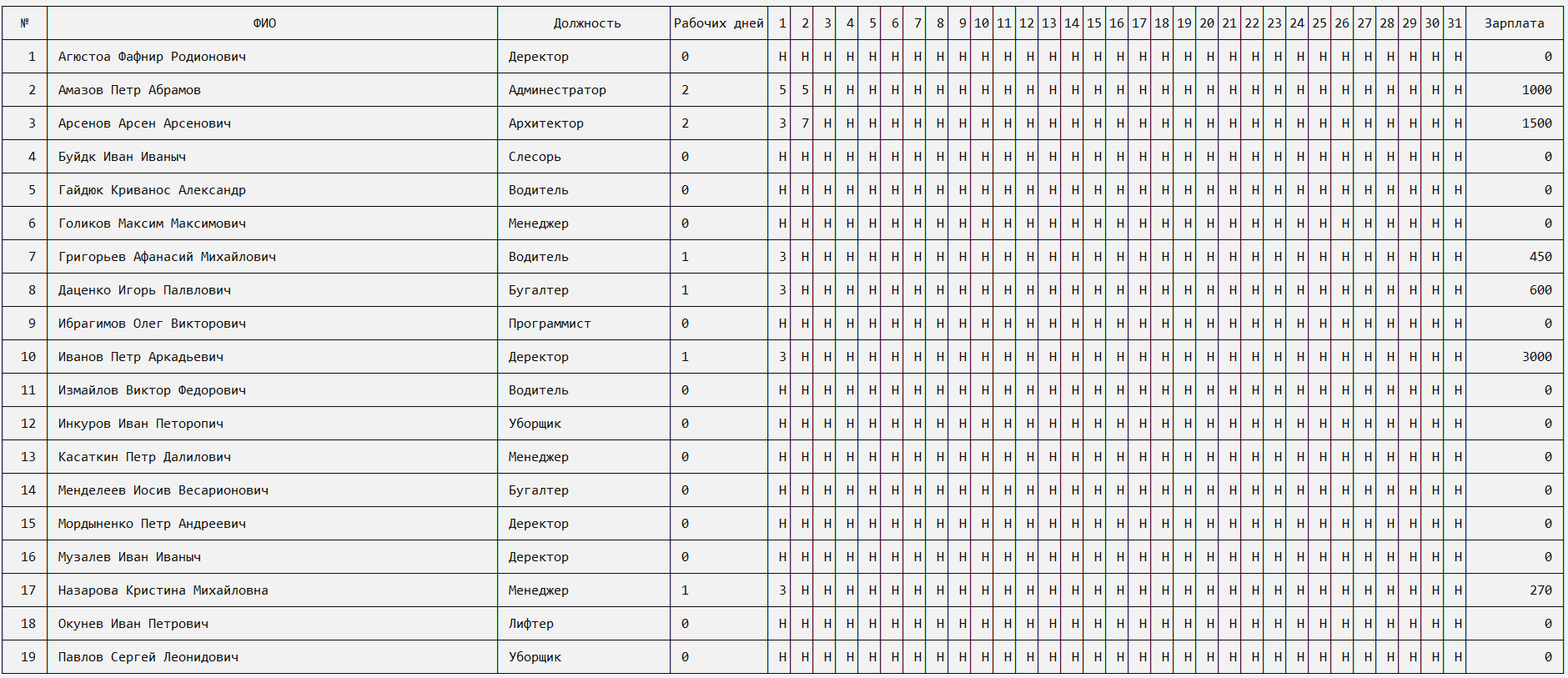
Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 3



5)Чтобы заполнить посещаемость введите: 1

Для записи нового сотрудника введите: 2

Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 4

Перезапись прошла успешно, данные за прошлый месяц хранятся в файле "Text\_copi"

Для продолжения нажмите любую клавишу . . .

Чтобы заполнить посещаемость введите: 1

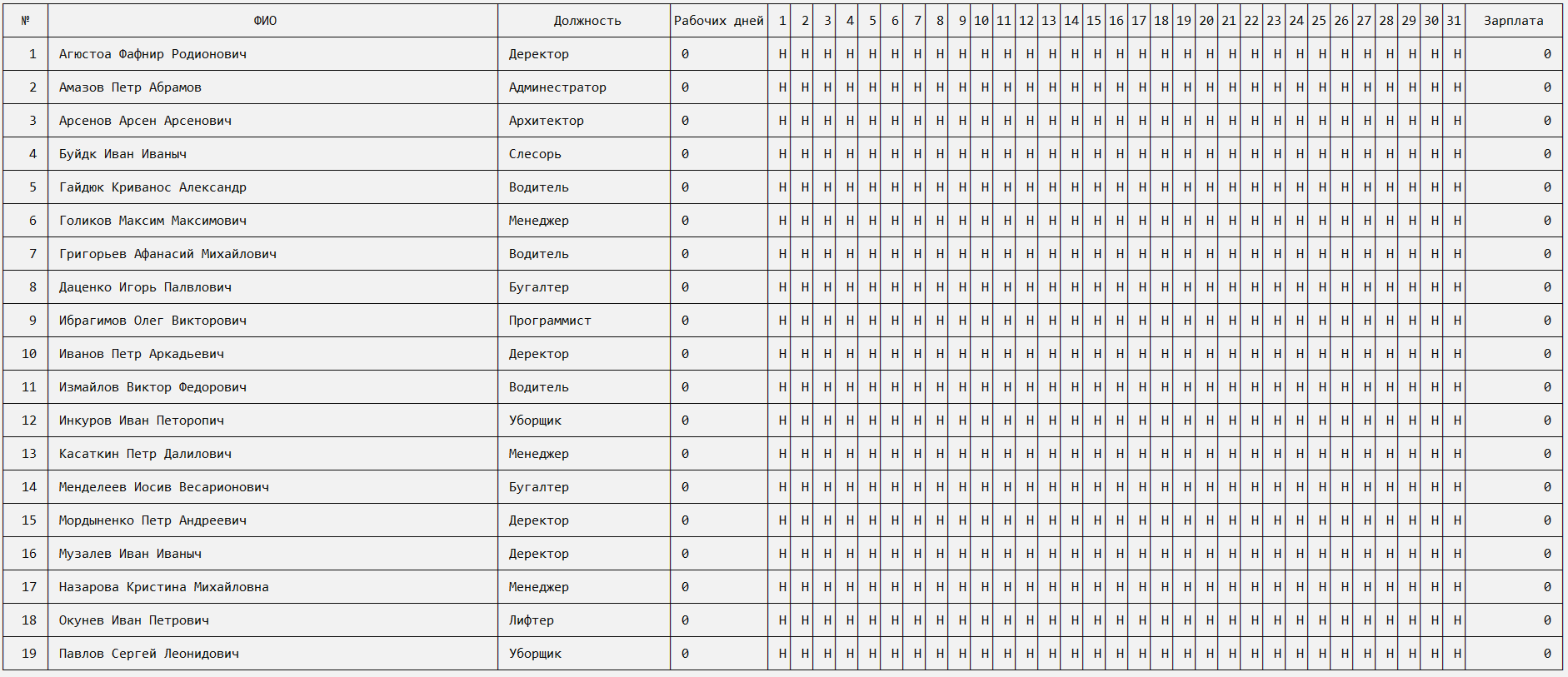
Для записи нового сотрудника введите: 2

Чтобы составить ведомость зарплаты введите: 3

Чтобы создать копию данных и обнулить текущие введите: 4

Чтобы удалить сотрудника из базы: 5

Выход: 0, или закройте консоль

Код действия: 3

# 4. Вывод

В данной работе были определены функциональные требования к системе средствами IDF0, составлено техническое задание, по нему смоделированы процессы, которые должны происходить в информационной системе, и реализованы в программном коде на языке «С++».

В работе в качестве базы данных использовались бинарные файлы, функции для работы с бинарными фалами использовались при реализации авторизации пользователя.

# Приложение А. Исходные коды информационной системы поддержки **бухгалтерии предприятия**

#include <iostream>

#include <fstream>

#include <string>

#include <windows.h>

#include <iomanip>

//1. Сформулировать функциональные и нефункциональные требования к информационной файловой системе своего варианта.

//2. Спроектировать структуры данных для заданной предметной области.

//3. Реализовать информационную систему средствами С++.

//Предметная область - бухгалтерия предприятия.

//Решаемые задачи :

//• учет труда и зарплаты работников;

//• табель учета рабочего времени составляется по подразделениям и включает\

ФИО работника, должность, количество отработанных часов и количество нерабочих дней за месяц;

//• формирование ведомостей на выплату зарплаты.

const char FNAME[] = "maillist"; //файл с данными

const char FNAME2[] = "maillist2"; //файл с данными

const char FNAME\_сегодня[] = "Text1"; //индефекаторы присутствующих сотрудников

const char FNAME\_копия[] = "Text\_copi"; //старая копия

//основные возможности

void Запись\_нового\_сотрудника(struct структура\* работник, int N, const char FNAME[]);

void Заполнение\_посищаемости(struct структура\* работник, int N, const char FNAME[], const char FNAME1[]);

void Состовление\_ведомости(struct структура\* работник, int N, const char FNAME[]);

void Подразделение(const char FNAME[]);

void перезапись(struct структура\* работник, int N, const char FNAME[]);

void удаление(struct структура\* работник, int N, const char FNAME[]);

using namespace std;

int max\_должность = 20; //длинна самой длнной должности

int max\_ФИО = 57; //длина фио

struct структура

{

char ФИО[3][19];

int должность;

int код;

int часы[31] = { 0 };

};

string Должности[12] = {

"Админестратор",

"Архитектор",

"Бугалтер",

"Водитель",

"Генеральный деректор",

"Деректор",

"Лифтер",

"Менеджер",

"Программист",

"Пилот",

"Слесорь",

"Уборщик" };

int main()

{

//подключение русского языка

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("color F0"); //фон белый буквы черные

int номер;

cout << "Введите номер подразделения: ";

cin >> номер;

if (номер == 1)

Подразделение(FNAME);

else if (номер == 2)

Подразделение(FNAME2);

}

/\* Сохранение списка. \*/

void запись\_структуры(struct структура\* работник, int N, const char FNAME[])

{

FILE\* fp;

register int i;

if ((fp = fopen(FNAME, "wb")) == NULL) {

printf("Ошибка при открытии файла.\n");

return;

}

fwrite(&N, sizeof(N), 1, fp);

for (i = 0; i < N; i++)

if (fwrite(&работник[i],

sizeof(struct структура), 1, fp) != 1)

printf("Ошибка при записи файла.\n");

fclose(fp);

}

/\* Загрузить файл. \*/

void чтение\_структуры(struct структура\* работник, int N, const char FNAME[])

{

FILE\* fp;

register int i;

if ((fp = fopen(FNAME, "rb")) == NULL) {

printf("Ошибка при открытии файла.\n");

return;

}

fread(&N, sizeof(N), 1, fp);

for (i = 0; i < N; i++)

if (fread(&работник[i],

sizeof(struct структура), 1, fp) != 1) {

if (feof(fp)) break;

printf("Ошибка при чтении файла.\n");

}

fclose(fp);

}

void Подразделение(const char FNAME[])

{

system("cls"); //отчистка экрана

int N, Команда = 1;

string код;

cout << "Введите пароль: ";

cin >> код;

if (код == "3")

while (Команда)

{

FILE\* fp = fopen(FNAME, "rb");

fread(&N, sizeof(N), 1, fp); fclose(fp);

//создание структуры

структура\* работник = new структура[N];

чтение\_структуры(работник, N, FNAME);

//чтение\_структуры(работник, FNAME, FNAME2);

system("cls"); //отчистка экрана

cout << "Чтобы заполнить посящаемость введите: 1";

cout << "\nДля записи нового сотрудника введите: 2";

cout << "\nЧтобы составить ведомость зарплаты введите: 3";

cout << "\nЧтобы создать копию данных и обнулить текущие введите: 4";

cout << "\nЧтобы удалить сотрудника из базы: 5";

cout << "\nВыход: 0, или закройте кансоль";

cout << "\n\nКод действия: ";

cin >> Команда;

if (Команда == 1)

Заполнение\_посищаемости(работник, N, FNAME, FNAME\_сегодня);

else if (Команда == 2)

Запись\_нового\_сотрудника(работник, N, FNAME);

else if (Команда == 3)

Состовление\_ведомости(работник, N, FNAME);

else if (Команда == 4)

перезапись(работник, N, FNAME);

else if (Команда == 5)

удаление(работник, N, FNAME);

//запись\_структуры(работник, N, FNAME);

delete[] работник;

}

}

void Сортировка(struct структура\* работник, int N, const char FNAME[]) {

string Min; //минимальный элемент

int jMin; //индекс минимального элемента

int iSort = 0; //граница отсортированной области

int j, i; //индексная переменная

структура Temp; //обменная переменная

string ФИО; //обменная переменная

//--------------Сортировка по именам---------------------

for (iSort = 0; iSort < N - 1; iSort++)

{

ФИО.clear();

for (i = 0; i < 3; i++)

ФИО += работник[iSort].ФИО[i];

//первый элемент из неупорядоченных назначаем минимальным

Min = ФИО; //минимум

jMin = iSort; //его индекс

//ищем минимальный элемент в оставшейся части массива

for (j = iSort + 1; j < N; j++)

{

ФИО.clear();

for (i = 0; i < 3; i++)

ФИО += работник[j].ФИО[i];

if (ФИО < Min) //очередной кандидат на минимальный

{

//запоминаем минимальный элемент и его номер

Min = ФИО;

jMin = j;

}//if

}//for j

//нашли минимум в неупорядоченной части массива

//ставим его на место первого в неупорядоченной части массива

//меняем элементы местами

Temp = работник[iSort];

работник[iSort] = работник[jMin];

работник[jMin] = Temp;

} //for iSort

}

void Запись\_нового\_сотрудника(struct структура\* работник, int N, const char FNAME[]) {

system("cls"); //отчистка экрана

int Код\_сотрудника;

int t = 0;

структура новый\_сотрудник;

cout << "Введите ФИО сотрудника: ";

cin >> новый\_сотрудник.ФИО[0] >> новый\_сотрудник.ФИО[1] >> новый\_сотрудник.ФИО[2];

system("cls"); //отчистка экрана

cout << "Введите код должности сотрудника из представленных ";

//ввод должности сотрудника

while (t < 1 || t > 12)

{

system("cls"); //отчистка экрана

//печать возможных должностей

for (int i = 0; i < 12; i++)

cout << "\n" << Должности[i] << ": " << i + 1;

cout << "\n\nКод должности сотрудника: ";

cin >> t;

//защита от дураков

if (t < 1 || t > 12)

{

cout << "\nНет такой должности\n";

system("PAUSE");

}

}

новый\_сотрудник.должность = t;

system("cls"); //отчистка экрана

//ввод личного кода сотрудника

t = 0;

do {

if (t)

cout << "\nсотрудник с таким кодом уже существует\n";

cout << "Персональный код сотрудника: ";

cin >> новый\_сотрудник.код;

t = 0;

//код уже существует

for (int i = 0; i < N; i++)

{

if (работник[i].код == новый\_сотрудник.код)

t = 1;

}

} while (t);

//создали новый масив структур работников

//переписали в него старый а в конец

//нового сотрудника

структура\* работник2 = new структура[N + 1];

for (int i = 0; i < N; i++)

{

работник2[i] = работник[i];

}работник2[N] = новый\_сотрудник;

Сортировка(работник2, N + 1, FNAME);

запись\_структуры(работник2, N + 1, FNAME);

delete[] работник2;

}

void Кто\_был(const char FNAME1[]) {

FILE\* fp = fopen(FNAME1, "wb");

int Код\_сотрудника = -1;

int число = 32;

system("cls"); //отчистка экрана

//ввод числа

while (число < 1 || число > 31)

{

cout << "Введите сегодняшнее число: ";

cin >> число;

//защита от дураков

if (число < 1 || число > 31) {

system("cls");

cout << "Неверный ввод числа\n";

}

else

fwrite(&число, sizeof(int), 1, fp);

}

//пока не введут 0

while (Код\_сотрудника != 0)

{

system("cls"); //отчистка экрана

cout << "Введите свой персональный код: ";

cin >> Код\_сотрудника;

fwrite(&Код\_сотрудника, sizeof(Код\_сотрудника), 1, fp);

}

fclose(fp);

system("cls"); //отчистка экрана

}

void Заполнение\_посищаемости(struct структура\* работник, int N, const char FNAME[], const char FNAME1[]) {

system("cls"); //отчистка экрана

int Часы\_работы;

int Число;

int N1 = 0; //сколько людей записалось

int j, i;

Кто\_был(FNAME1);

//считаю количество записавшихся

FILE\* in1 = fopen(FNAME1, "rb");

fread(&Число, sizeof(int), 1, in1);//чтение числа

while (fread(&i, sizeof(int), 1, in1) == 1) {

N1++;

}fclose(in1);

//чтение принятых индификаторов присуствия

FILE\* out1 = fopen(FNAME1, "rb");

int\* ЛКод = new int[N1];

fread(&i, sizeof(int), 1, out1);//уход с 1-го элемента

for (i = 0; i < N1; i++)

fread(&ЛКод[i], sizeof(int), 1, out1);

fclose(out1);

//создание и обнуление флагов присуствия

bool\* f = new bool[N];

for (j = 0; j < N; j++)

f[j] = 0;

//сравнение присутствуюших с базай

for (j = 0; j < N; j++)

for (i = 0; i < N1; i++)

if (ЛКод[i] == работник[j].код)

f[j] = 1;

//дополнение данных

for (j = 0; j < N; j++) {

if (f[j]) {

Часы\_работы = -1;//опустить флаг

//ввод часов работы

while (Часы\_работы < 0 || Часы\_работы > 24)

{

for (int i = 0; i < 3; i++)

cout << работник[j].ФИО[i] << " ";

cout << "\nОн сегодня работал: ";

cin >> Часы\_работы;

//защита от дурака

if (Часы\_работы < 0 || Часы\_работы > 24) {

system("cls");

cout << "Он не мог столько работать\n";

}

else

работник[j].часы[Число - 1] = Часы\_работы;

}

}

system("cls"); //отчистко экрана

}

запись\_структуры(работник, N, FNAME);

//запись\_структуры(работник, N, FNAME, FNAME2);

delete[] ЛКод; delete[] f;

}

//подключение английского языка

void a() {

SetConsoleCP(866);

SetConsoleOutputCP(866);

}

//подключение русского языка

void r() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

}

void шапка(int max, int max\_должность) {

//начало шапки

a();

cout << char(218) << setfill(char(196)) << setw(6);

cout << char(194) << setfill(char(196)) << setw(max + 3);

cout << char(194) << setfill(char(196)) << setw(max\_должность + 3);

cout << char(194) << setfill(char(196)) << setw(13);

for (size\_t m = 0; m < 31; m++)

cout << char(194) << setfill(char(196)) << setw(3);

cout << char(194) << setfill(char(196)) << setw(13);

cout << char(191);

cout << "\n";

//шапка номера

cout << char(179);

r(); cout << " № "; a();

//шапка ФИО

int i = (max + 3) / 2;

cout << char(179);

r(); cout << setfill(char(32)) << setw(i) << " ФИО"; a();

if ((max + 3) % 2 == 1)

i += 1;

cout << setfill(char(32)) << setw(i);

//шапка должности

cout << char(179);

r(); cout << " Должность "; a();

//шапка рабочих дней

cout << char(179);

r(); cout << "Рабочих дней"; a();

//шапки часов работ

for (size\_t m = 0; m < 31; m++)

cout << char(179) << setfill(char(32)) << setw(2) << m + 1;

//шапка зарплаты

cout << char(179);

r(); cout << " Зарплата "; a();

cout << char(179);

cout << "\n";

}

void отчерк() {

cout << char(195) << setfill(char(196)) << setw(6); //номер

cout << char(197) << setfill(char(196)) << setw(max\_ФИО + 3); //ФИО

cout << char(197) << setfill(char(196)) << setw(max\_должность + 3); //должность

cout << char(197) << setfill(char(196)) << setw(13); //рабочие дни

for (size\_t m = 0; m < 31; m++) //часы работ

cout << char(197) << setfill(char(196)) << setw(3);

cout << char(197) << setfill(char(196)) << setw(13) << char(180); //зарплата

cout << "\n";

}

void серкдина(int N, string ФИО, int Должность, int Час[31], int колличество\_рабочих\_дней, int колличество\_отработанных\_часов) {

int Зарплата[12] = { 100,150,200,150,10000,1000,80,90,5000,1200,100,50 };

cout << char(179) << " " << setfill(char(32)) << setw(3) << N << " ";

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(max\_ФИО) << ФИО << " " << right; a();

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(max\_должность) << Должности[Должность - 1] << " " << right; a();

cout << char(179);

r(); cout << " " << left << setfill(char(32)) << setw(10) << колличество\_рабочих\_дней << " " << right; a();

for (size\_t i = 0; i < 31; i++)

{

if (Час[i])

cout << char(179) << setfill(char(32)) << setw(2) << Час[i];

else

cout << char(179) << " H";

}

cout << char(179);

cout << " " << setfill(char(32)) << setw(10) << колличество\_отработанных\_часов \* Зарплата[Должность - 1] << " ";

cout << char(179) << "\n";

}

void конец() {

cout << char(192) << setfill(char(196)) << setw(6); //номер

cout << char(193) << setfill(char(196)) << setw(max\_ФИО + 3); //ФИО

cout << char(193) << setfill(char(196)) << setw(max\_должность + 3); //должность

cout << char(193) << setfill(char(196)) << setw(13); //рабочие дни

for (size\_t m = 0; m < 31; m++) //часы работ

cout << char(193) << setfill(char(196)) << setw(3);

cout << char(193) << setfill(char(196)) << setw(13) << char(217); //зарплата

cout << "\n\n\n"; r();

}

void Состовление\_ведомости(struct структура\* работник, int N, const char FNAME[]) {

int Должность;

int i, j;

string ФИО;

system("cls");

шапка(max\_ФИО, max\_должность); //печать начала таблици

for (i = 0; i < N; i++)

{

for (j = 0; j < 3; j++) {

ФИО.append(работник[i].ФИО[j]); //дописать в конец ФИО

ФИО.push\_back(' ');//дописать пробел

}

//считаю колличество рабочих дней

int колличество\_рабочих\_дней = 0;

for (size\_t m = 0; m < 31; m++)

if (работник[i].часы[m])

колличество\_рабочих\_дней++;

int колличество\_отработанных\_часов = 0;

for (size\_t m = 0; m < 31; m++)

колличество\_отработанных\_часов += работник[i].часы[m];

отчерк(); //рисую отчерк

серкдина(i + 1, ФИО, работник[i].должность, работник[i].часы, колличество\_рабочих\_дней, колличество\_отработанных\_часов);

ФИО.clear(); //отчистка буферов

}

конец();

system("PAUSE");

}

void перезапись(struct структура\* работник, int N, const char FNAME[])

{

system("cls"); //отчистка экрана

char s[256]; //буфер

запись\_структуры(работник, N, FNAME\_копия);

//обнуление часов

for (int j = 0; j < N; j++)

for (int i = 0; i < 31; i++)

работник[j].часы[i] = 0;

запись\_структуры(работник, N, FNAME);

cout << "Перезапись прошла успешно, данные за прошлый месяц хранятся в файле \"Text\_copi\"\n";

system("PAUSE");

}

void удаление(struct структура\* работник, int N, const char FNAME[])

{

int уволеный;

system("cls"); //отчистка экрана

cout << "Введите порядковый номер уволившегося сотрудника: ";

cin >> уволеный;

if (уволеный > N || уволеный < 1)

{

cout << "\nНет такого сотрудника\n"; system("PAUSE");

return;

}

for (int i = 0; i < max\_ФИО / 3; i++)

работник[уволеный - 1].ФИО[0][i] = 'я';

Сортировка(работник, N, FNAME);

запись\_структуры(работник, N - 1, FNAME);

}