**Описание структур данных**

Хранение данных программы осуществляется в форме линейного однонаправленного списка. Каждый элемент списка представляет собой структуру (см. рис. 3), состоящую из полей с данными, а также указателя на соседний (следующий) элемент списка.

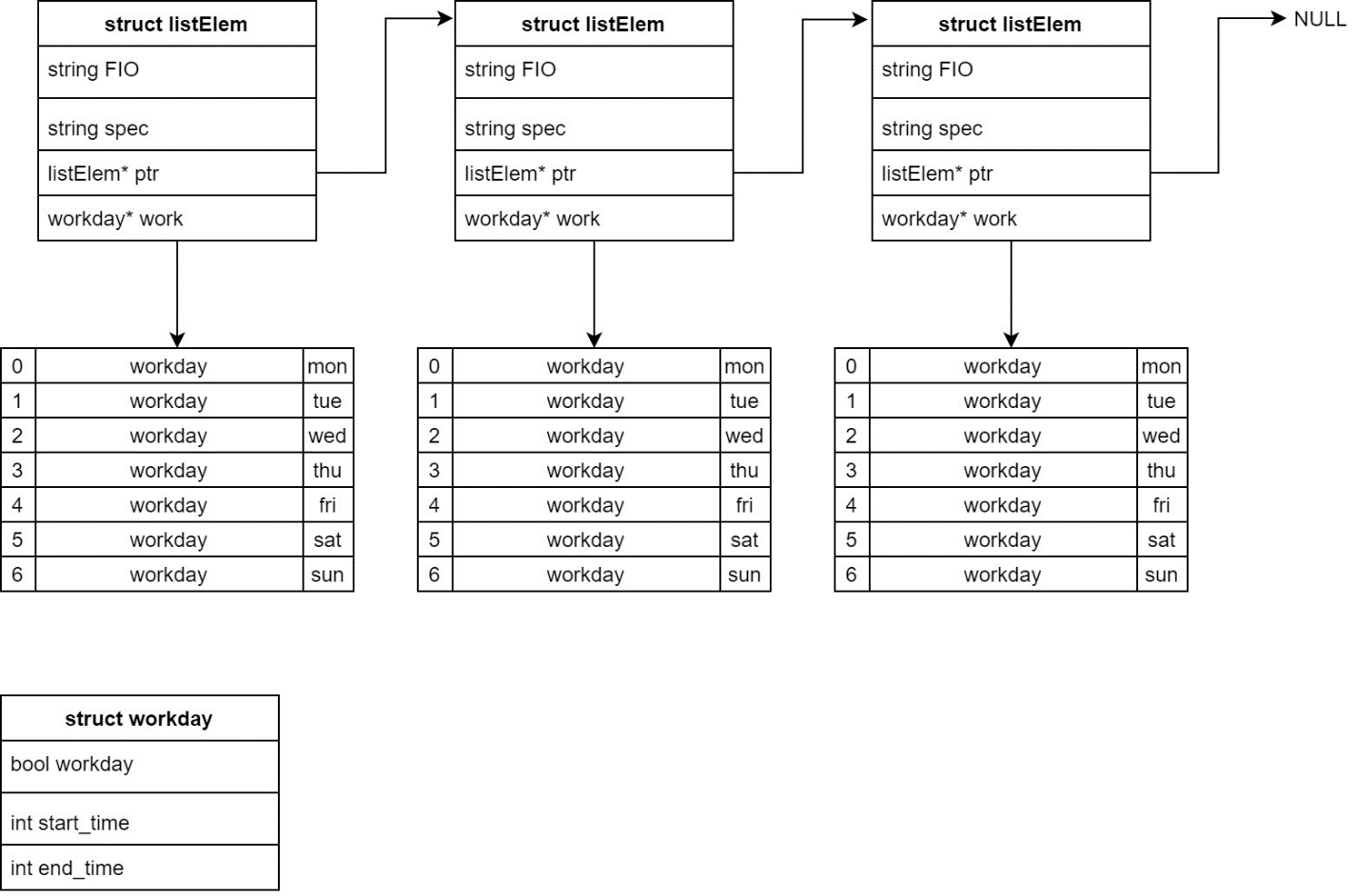


Рис. 3 - Структура храпения данных

Структуры

Врач list представлен в памяти программы в виде структуры с следующими полями:

string FIO, spec - ФИО и специализация

workday\* work - Указатель на массив рабочих дней

listElem\* ptr=NULL - Указатель на следующий элемент

Структура для хранения данных по рабочему дню workday содержит поля:

bool workday - Рабочий ли день

int start\_time, end\_time - Время начала и конца рабочего дня

**Описание пользовательских функций**

Модуль functions

**listElem\* init**

Назначение: Функция инициализации 0-го узла списка.

Параметры: **string FIO, string spec, workday\* work**

Принимает ФИО врача, специальность и массив рабочих дней.

Возвращаемое значение: **listElem\* -** указатель на начальный элемент списка

**add\_new\_item**

Назначение: Функция добавления нового узла в список

Параметры: **listElem\* root, string FIO, string spec, workday\* work**

Принимает указатель на начальный элемент списка, ФИО врача, специальность и массив рабочих дней.

Возвращаемое значение: -

**listprint**

Назначение: Функция поэлементного вывода на экран всего списка.

Параметры: **listElem\* root**

Принимает указатель на начальный элемент списка.

Возвращаемое значение: -

**Get\_new\_item\_from\_keyboard**

Назначение: Функция для ввода нового элемента с клавиатуры

Параметры: **listElem\*\* root**

Принимает указатель на указатель на начальный элемент списка.

Возвращаемое значение: -

**save**

Назначение: Функция для сохранения списка в файл

Параметры: **listElem\* root**

Принимает указатель на начальный элемент списка.

Возвращаемое значение: -

**read**

Назначение: Функция для чтения списка из файла

Параметры: **listElem\*\* root**

Принимает указатель на указатель на начальный элемент списка.

Возвращаемое значение: -

**delete\_item**

Назначение: Функция удаления элемента списка

Параметры: **listElem\* lst, listElem\*\* root**

Принимает указатель на удаляемый элемент и указатель на указатель на начало списка.

Возвращаемое значение: -

**OnlyOneElementPrint**

Назначение: Функция печати одного элемента списка

Параметры: **listElem\* elem**

Принимает указатель на элемент списка для печати.

Возвращаемое значение: -

**find**

Назначение: Функция поиска элемента списка по ФИО врача.

Принимает указатель на начало списка и искомое ФИО. Возвращает указатель на найденный элемент.

Параметры: **listElem\* root, string FIOfind**

Принимает указатель на начальный элемент списка и ФИО искомого врача

Возвращаемое значение: **listElem\***

Указатель на найденный элемент.

**findname**

Назначение: Функция обработки запроса от пользователя поиска элемента списка по ФИО врача.

Принимает указатель на начало списка.

Параметры: **listElem \*root**

Принимает указатель на начальный элемент списка.

Возвращаемое значение: -

**deletename**

Назначение: Функция обработки запроса от пользователя на удаление элемента списка по ФИО врача.

Принимает указатель на указатель на начало списка.

Параметры: **listElem\*\* root**

Принимает указатель на указатель на начальный элемент списка.

Возвращаемое значение: -

**sortbyname**

Назначение: Функция сортировки элементов списка по алфавитному порядку ФИО врачей.

Параметры: **listElem\*\* root**

Принимает указатель на указатель на начальный элемент списка.

Возвращаемое значение: -

**editname**

Назначение: Функция обработки запроса от пользователя на редактирование элемента списка по ФИО врача.

Параметры: **listElem\* root**

Принимает указатель на начало списка.

Возвращаемое значение: -

**Алгоритм программы**

Модуль functions

listElem\* init(string FIO, string spec, workday\* work)

listElem\* root

root = new listElem

root->FIO = FIO

root->work = work

root->spec = spec

root->ptr = NULL

return root

void add\_new\_item(listElem\* root, string FIO, string spec, workday\* work)

listElem \*temp,\*p

temp = new listElem

Пока (root->ptr != NULL)

root = root->ptr

p = root->ptr

root->ptr = temp

temp->FIO = FIO

temp->work = work

temp->spec = spec

temp->ptr = p

void delete\_item(listElem\* elem, listElem\*\* root)

listElem\* temp

Если (elem != NULL)

Если (elem != \*root)

temp = \*root

Пока (temp->ptr != elem)

temp = temp->ptr

temp->ptr = elem->ptr

Иначе

\*root = elem->ptr

delete elem

void listprint(listElem\* root)

listElem\* p

p = root

Пока (p != NULL)

вывод << p->FIO << " " << p->spec << endl

Цикл по i от 0 до 7

вывод << "В день недели " << i << " "

Если (p->work[i].workday)

вывод << "рабочий день от " << p->work[i].start\_time << " до " << p->work[i].end\_time << endl

Иначе

вывод << "выходной" << endl

p = p->ptr

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

void OnlyOneElementPrint(listElem\* elem)

listElem\* p

p = elem

Если (p != NULL)

вывод << p->FIO << " " << p->spec << endl

Цикл по i от 0 до 7

вывод << "В день недели " << i << " "

Если (p->work[i].workday)

вывод << "рабочий день от " << p->work[i].start\_time << " до " << p->work[i].end\_time << endl

Иначе

вывод << "выходной" << endl

Иначе

вывод << "NULL" << endl

listElem\* find(listElem\* root, string FIOfind)

listElem\* p

p = root

Пока (p != NULL)

Если (p->FIO == FIOfind)

return p

p = p->ptr

return NULL

void findname(listElem \*root)

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

string FIOfind

вывод << "Какое ФИО ищем:"

getline(ввод, FIOfind)

OnlyOneElementPrint(find(root, FIOfind))

void sortbyname(listElem\*\* root)

listElem \*NewList=NULL

listElem \*p, \*minimum

Пока (\*root != NULL)

p = \*root

minimum = p

Пока (p != NULL)

Если (p->FIO < minimum->FIO)

minimum = p

p = p->ptr

Если (NewList == NULL)

NewList = init(minimum->FIO, minimum->spec, minimum->work)

Иначе

p = NewList

Пока (p->ptr != NULL)

p = p->ptr

add\_new\_item(p, minimum->FIO, minimum->spec, minimum->work)

delete\_item(minimum, root)

\*root = NewList

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

void deletename(listElem\*\* root)

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

string FIOfind

вывод << "Какое ФИО ищем:"

getline(ввод, FIOfind)

delete\_item(find(\*root, FIOfind), root)

void editname(listElem\* root)

listElem\* edit = NULL

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

string FIOfind

вывод << "Какое ФИО ищем:"

getline(ввод, FIOfind)

edit=find(root, FIOfind)

Если (edit != NULL)

OnlyOneElementPrint(edit)

вывод << endl << "Новое ФИО врача:"

getline(ввод, edit->FIO)

вывод << endl << "Новая специальность:"

getline(ввод, edit->spec)

Цикл по i от 0 до 7

вывод << "Во сколько начинает работать врач в " << i << "-й день недели? ('-1' - если выходной):" << endl

ввод >> edit->work[i].start\_time

Если (edit->work[i].start\_time == -1)

edit->work[i].workday = false

Иначе

edit->work[i].workday = true

вывод << "Во сколько заканчивает работать врач в " << i << "-й день недели?" << endl

ввод >> edit->work[i].end\_time

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

void Get\_new\_item\_from\_keyboard(listElem\*\* root)

string FIO, spec

workday\* work

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

вывод << endl << "ФИО врача:"

getline(ввод, FIO)

вывод << endl << "Специальность:"

getline(ввод, spec)

work = new workday[7]

Цикл по i от 0 до 7

вывод << "Во сколько начинает работать врач в " << i << "-й день недели? ('-1' - если выходной):" << endl

ввод >> work[i].start\_time

Если (work[i].start\_time == -1)

work[i].workday = false

Иначе

work[i].workday = true

вывод << "Во сколько заканчивает работать врач в " << i << "-й день недели?" << endl

ввод >> work[i].end\_time

Если (\*root == NULL)

\*root = init(FIO, spec, work)

Иначе

add\_new\_item(\*root, FIO, spec, work)

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

void save(listElem\* root)

string output\_file\_name

вывод << "output filename:"

ввод >> output\_file\_name

ofstream out(output\_file\_name)

Если (!out)

вывод << "Can't open file"

return

listElem\* p

p = root

Пока (p != NULL)

out << p->FIO << endl << p->spec << endl

Цикл по i от 0 до 7

Если (p->work[i].workday)

out << p->work[i].start\_time << endl << p->work[i].end\_time << endl

Иначе

out << "-1" << endl

p = p->ptr

out.close()

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

void read(listElem\*\* root)

string input\_file\_name, FIO, spec

workday\* work

int starttime

int i = 0, k

вывод << "input filename:"

ввод >> input\_file\_name

Если stream fs(input\_file\_name)

Если (!fs)

вывод << "Can't open file"

return

Пока (fs.good())

Если (getline(fs, FIO))

getline(fs, spec)

work = new workday[7]

Цикл по i от 0 до 7

fs >> work[i].start\_time

Если (work[i].start\_time != -1)

work[i].workday = true

fs >> work[i].end\_time

Иначе

work[i].workday = false

Если (\*root == NULL)

\*root = init(FIO, spec, work)

Иначе

add\_new\_item(\*root, FIO, spec, work)

getline(fs, input\_file\_name)

ввод.clear()

ввод.ignore(numeric\_limits<streamsize>::max(), '\n')

fs.close()

main

listElem\* root = NULL

char ch

SetConsoleCP(1251)

SetConsoleOutputCP(1251)

Пока (true)

вывод << "1 - Добавить новый элемент с клавиатуры" << endl << "2 - Вывод на экран" << endl << "3 - загрузить из файла" << endl << "4 - сохранить в файл" << endl

вывод << "5 - Удалить элемент по ФИО" << endl << "6 - Отредактировать элемент по ФИО" << endl << "7 - Найти врача по ФИО" << endl << "8 - Отсортировать врачей по ФИО" << endl

Если (!ввод.get(ch)) break

Переключатель по (ch)

В случае '1':

Get\_new\_item\_from\_keyboard(&root)

break

В случае '2':

listprint(root)

break

В случае '3':

read(&root)

break

В случае '4':

save(root)

break

В случае '5':

deletename(&root)

break

В случае '6':

editname(root)

break

В случае '7':

findname(root)

break

В случае '8':

sortbyname(&root)

break

return 0