**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования «Московский авиационный институт (национальный исследовательский университет)»**

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра №316 «Системное моделирование и автоматизированное проектирование»

Курсовая работа

По теме: «Разработка программы журнал посещения занятий»

По курсу: «Технология разработки программного обеспечения»

Выполнил: Масликова Е.П.

Студент группы: М3О-218М-18

Проверил: Бутко А.О.

Москва 2019

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 3

1. Создание консольной программы 4

2. Создание windows приложения 6

3. Описание жизненного цикла 11

3.1. Описание краткого жизненного цикла 11

3.2. Описание полного жизненного цикла 11

4. Описание модели проектирования программы 14

ВЫВОДЫ 18

**ВВЕДЕНИЕ**

Разработка программного обеспечения — деятельность по созданию нового [программного обеспечения](https://ru.wikipedia.org/wiki/Программное_обеспечение).

В современных условиях важной областью стало информационное обеспечение, которое состоит в сборе и переработке информации, необходимой для принятия обоснованных управленческих решений. Передача информации о положении и деятельности предприятия на высший уровень управления и взаимный обмен информацией между всеми подразделениями организации осуществляются на базе современной электронно-вычислительной техники и других технических средствах связи.

По мере своего дальнейшего развития административные системы обработки данных перерастают в автоматизированные системы управления соответствующими объектами, в которых, как правило, не ограничиваются одной ЭВМ, а в составе двух и более ЭВМ объединяют в вычислительный комплекс.

**1. Создание консольной программы**

Результат работы консольной программы «Журнал посещения занятий» представлен на рисунке 1.



Рис 1. Консольная программа «Журнал посещения занятий»

Код разработанной программы на языке программирования Python3:

import json

import datetime

import pandas as pd

def get\_config(file\_name):

file = open(file\_name, 'r', encoding='utf-8')

result = json.load(file)

file.close()

return result

def get\_valid\_visit(name):

while True:

result = str(input(name + ': '))

if result == '+' or result == '-':

return result == '+'

def write\_file(file\_name, content):

file = open(file\_name, 'w')

json.dump(content, file, indent=4, ensure\_ascii=False)

file.close()

config = get\_config('math.json')

now = datetime.datetime.now()

now = str(now.day) + '.' + str(now.month) + '.' + str(now.year)

print(now)

index = []

column = []

df = []

for i in range(len(config)):

index.append(config[i]['name'])

config[i]['visit'].append({

'date': now,

'presence': get\_valid\_visit(config[i]['name'])

})

\_df = []

for j in range(len(config[i]['visit'])):

\_df.append(config[i]['visit'][j]['presence'])

df.append(\_df)

for i in range(len(config[0]['visit'])):

column.append(config[0]['visit'][i]['date'])

df = pd.DataFrame(df, index = index, columns = column)

print(df)

write\_file('math.json', config)

**2. Создание windows приложения**

Результат работы программы представлен на рисунке 2.

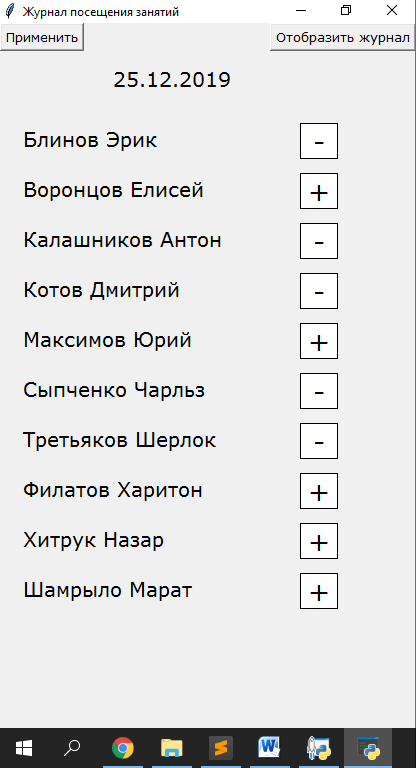


Рис 2. Windows приложение

Код разработанной программы на языке программирования Python3:

from tkinter import \*

import json

import datetime

import pandas as pd

class WindowApplication():

def \_\_init\_\_(self):

obj = self.get\_config('math.json')

self.labels = [''] \* (len(obj) + 1)

for i in range(len(self.labels)):

self.labels[i] = Label(root,

font = ('Verdana', 15),

justify = LEFT

)

now = datetime.datetime.now()

now = str(now.day) + '.' + str(now.month) + '.' + str(now.year)

self.labels[0]['text'] = now

self.labels[0].place(x = 110, y = 40)

for i in range(len(obj)):

self.labels[i + 1]['text'] = obj[i]['name']

self.labels[i + 1].place(x = 20, y = 50 \* (i + 2))

self.entry = [''] \* len(obj)

for i in range(len(self.entry)):

self.entry[i] = Entry(

font = ('Verdana', 20),

justify = CENTER,

relief = SOLID,

width = 2

)

for i in range(len(obj)):

self.entry[i].place(x = 300, y = 50 \* (i + 2))

self.buttons = [''] \* 2

for i in range(len(self.buttons)):

self.buttons[i] = Button(root,

font = ('Verdana', 10)

)

self.buttons[0]['text'] = 'Применить'

self.buttons[1]['text'] = 'Отобразить журнал'

self.buttons[0].place(x = 0, y = 0)

self.buttons[1].place(x = 270, y = 0)

self.buttons[0].bind('<ButtonRelease-1>', self.main)

self.buttons[1].bind('<ButtonRelease-1>', self.show)

def get\_config(self, file\_name):

file = open(file\_name, 'r', encoding='utf-8')

result = json.load(file)

file.close()

return result

def write\_file(self, file\_name, content):

file = open(file\_name, 'w')

json.dump(content, file, indent=4, ensure\_ascii=False)

file.close()

def get\_valid\_visit(self, number):

return str(self.entry[number].get()) == '+'

def main(self, event):

config = self.get\_config('math.json')

now = datetime.datetime.now()

now = str(now.day) + '.' + str(now.month) + '.' + str(now.year)

#print(now)

for i in range(len(config)):

config[i]['visit'].append({

'date': now,

'presence': self.get\_valid\_visit(i)

})

self.write\_file('math.json', config)

def show(self, event):

config = self.get\_config('math.json')

index = []

column = []

df = []

for i in range(len(config)):

index.append(config[i]['name'])

\_df = []

for j in range(len(config[i]['visit'])):

\_df.append(config[i]['visit'][j]['presence'])

df.append(\_df)

for i in range(len(config[0]['visit'])):

column.append(config[0]['visit'][i]['date'])

df = pd.DataFrame(df, index = index, columns = column)

child = Tk()

child.title('Просмотр журнала')

child.wm\_geometry('1000x500')

child.resizable(width=False, height=False)

label = Label(child,

font = ('Verdana', 15),

justify = RIGHT

)

label['text'] = df

label.place(x = 0, y = 50)

child.mainloop()

root = Tk()

root.title('Журнал посещения занятий')

root.wm\_geometry('415x500')

root.resizable(width=False, height=True)

application = WindowApplication()

root.mainloop()

Дата посещения занятия студентами формируется в соответствии с текущим временем на компьютере, когда программа запускается. Пользователь ее вручную не указывает.

Все данные записываются в файл формата JSON.

В рамках выполнения курсовой работы данные в журнале посещений приведены за 24 и 25 декабря 2019 года.

Информация, содержащаяся в файле для указанного промежутка времени:

[

{

"name": "Блинов Эрик",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": false

}

]

},

{

"name": "Воронцов Елисей",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": true

}

]

},

{

"name": "Калашников Антон",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": false

}

]

},

{

"name": "Котов Дмитрий",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": false

}

]

},

{

"name": "Максимов Юрий",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": true

}

]

},

{

"name": "Сыпченко Чарльз",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": false

}

]

},

{

"name": "Третьяков Шерлок",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": false

}

]

},

{

"name": "Филатов Харитон",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": true

}

]

},

{

"name": "Хитрук Назар",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": true

}

]

},

{

"name": "Шамрыло Марат",

"visit": [

{

"date": "24.12.2019",

"presence": true

},

{

"date": "25.12.2019",

"presence": true

}

]

}

]

**3. Описание жизненного цикла**

**3.1. Описание краткого жизненного цикла**

На рисунке 3 приведена диаграмма, описывающая краткий жизненный цикл разработанной программы, который выражается в трех больших этапах.

Рис 3. Диаграмма краткого жизненного цикла

На этапе планирования определяют и координируют действия по разработке программной системы.

На этапе разработки осуществляется постановка задачи, корректирование, кодирование и получение исполняемого кода.

На интерферированном этапе происходит проверка, коррекция и определение полноты программного обеспечения, а так же его выпуск. Так же верификация и контроль за конфигурацией системы, оценка качества и проверка взаимодействия между этапами.

**3.2. Описание полного жизненного цикла**

Вариант описания краткого жизненного цикла не вполне учитывает отличительные особенности проектирования программ. Поэтому наиболее подходящим является жизненный цикл, приведенный на рисунке 4. Он называется полным жизненным циклом.

Рис 4. Диаграмма полного жизненного цикла

На этапе № 1 определяются необходимые действия, планы и организация управления проектом вцелом. Определяются меры по обеспечению непрерывного выполнения фаз жизненного цикла.

На этапе № 2 определяются общие характеристики систем без учета средств реализации, которым она должна удовлетворять. Устанавливаются основные требования к действиям, выполняемым системой.

На этапе № 3 описывается, как должны удовлетворять запросы пользователя в терминах конкретных функциональных понятий, описываются действия предполагаемой системы, хранимые данные, используемый интерфейс без учета физической реализации. Проверяется пригодность этих конкретных понятий.

На этапе № 4 устанавливается структура системы, ее архитектура в терминах основных компонент и их предполагаемой реализации. Устанавливаются требования для каждого компонента, а так же стратегия тестирования и интеграция.

На этапе № 5 определяются конкретные программные компоненты, которые будут разрабатываться и внедряться в конечную систему. Выполняется проверка этих компонент на соответствие общим требованиям к программному обеспечению и определение функциональных эксплуатационных и тестовых требований к каждому конкретному компоненту.

На этапе № 6 в терминах используемых программных конструкций производится описание того, как каждый компонент будет разрабатываться. Описываются режимы использования каждого компонента в системе.

На этапе № 7 происходит создание и тестирование отдельных модулей, документирование и приемка программных компонент, которые составляют программную систему.

На этапе № 8 происходит тестирование работоспособности и функциональной законченности программных частей системы в представленном окружении.

На этапе № 9 осуществляется тестирование работоспособности и функциональной законченности частей общей системы вцелом.

На этапе № 10 происходит приемка системы заказчиком и ее поставка.

На этапе № 11 выпускаются последующие варианты или версии системы, необходимость в которых возникает из-за устранений дефектов или отработки измененных требований.

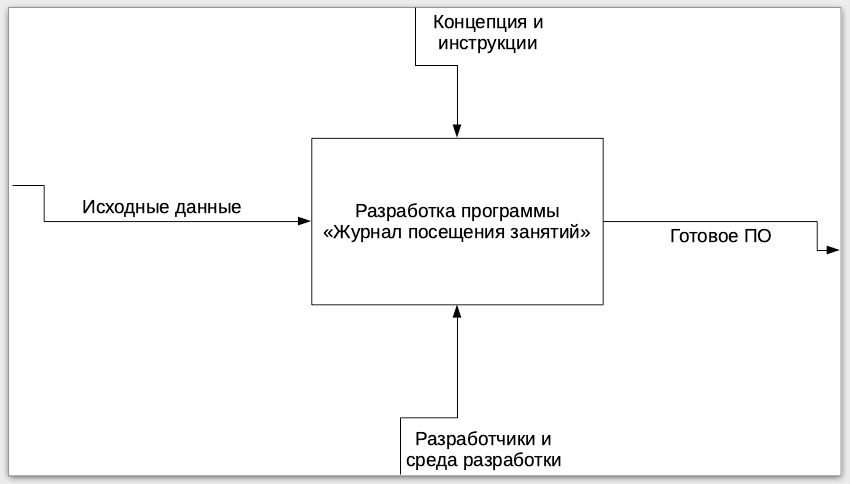
На этапе № 12 формируется окончательная модель проектных действий с анализом достоинств и недостатков и использование их в качестве основных для улучшения процесса разработки.

**4. Описание модели проектирования программы**

Разработка программы «Журнал посещения занятий» осуществляется на основе трех входных компонентов:

* Концепция и инструкции;
* Исходные данные;
* Разработчики и среда разработки.

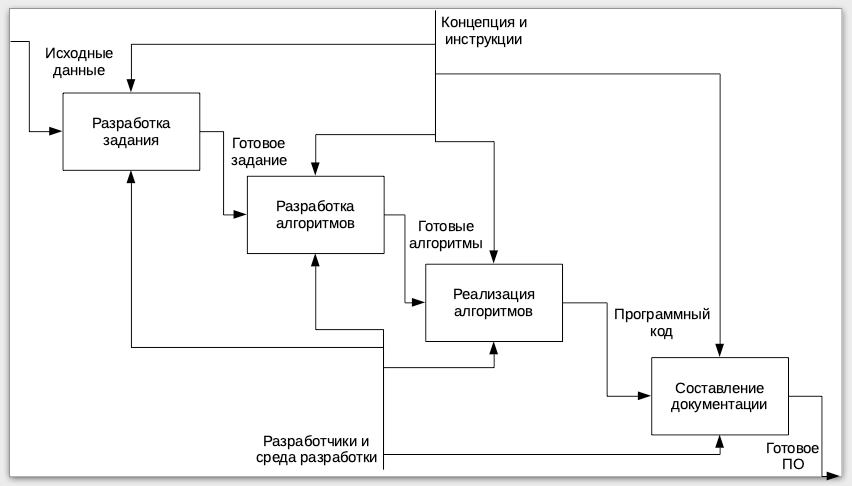
Результатом разработки является готовый программный продукт. На рисунке 5 приведено графическое представление.

Рис 5. Графическая схема разработки программы

Разработка программы может быть представлена как последовательность этапов:

* Разработка задания;
* Разработка алгоритмов;
* Реализация алгоритмов;
* Составление документации.

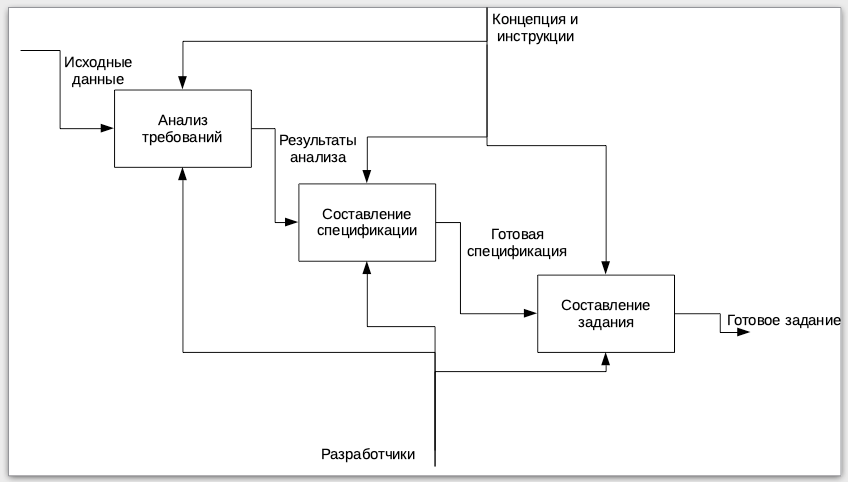
Графическое представление последовательно следующих друг за другом этапов разработки программы приведено на рисунке 6.

Рис 6. Последовательные этапы разработки программы

Разработка задания включает следующие этапы:

* Анализ требований;
* Составление спецификации;
* Составление задания.

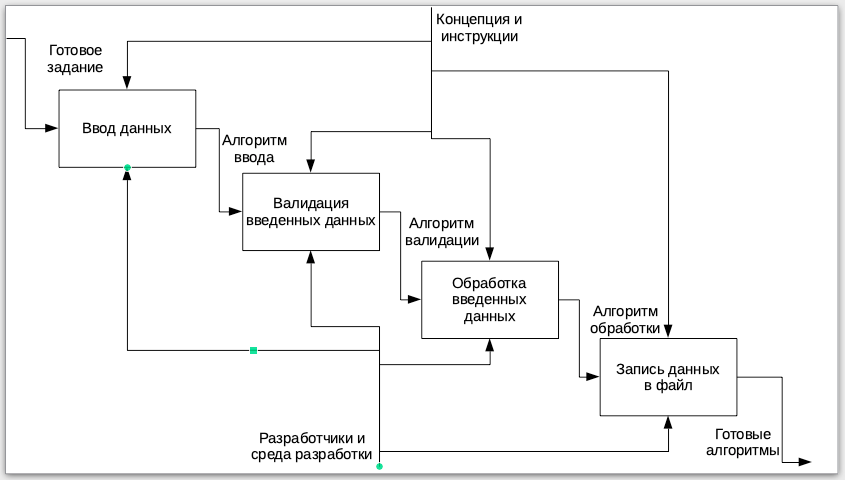
Результатом является готовое задание. На рисунке 7 приведены последовательные этапы работки задания.

Рис 7. Разработка задания программы «Журнал посещения занятий»

Разработка алгоритмов включает следующие этапы:

* Ввод исходных данных;
* Обработка исходных данных;
* Вывод результата пользователю.

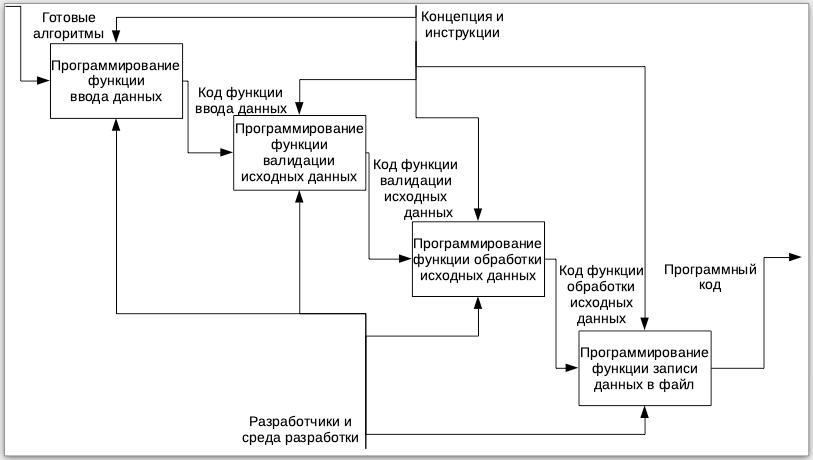
Результатом являются готовые алгоритмы. На рисунке 8 приведены последовательные этапы работки алгоритмов.

Рис 8. Разработка алгоритмов программы «Журнал посещения занятий»

Реализация алгоритмов включает следующие этапы:

* Программирование функции ввода данных;
* Программирование функции обработки данных;
* Программирование функции вывода результата.

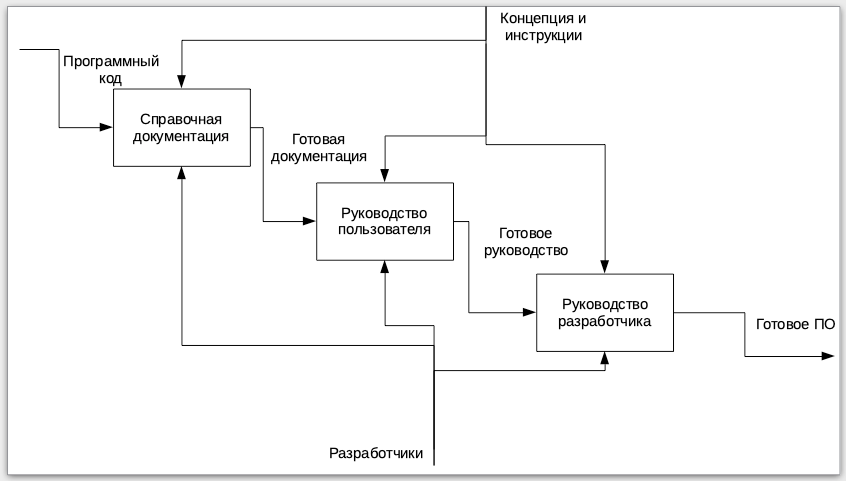
Результатом является программный код. На рисунке 9 приведены последовательные этапы реализации алгоритмов.

Рис 9. Реализация алгоритмов программы «Журнал посещения занятий»

Составление документации включает следующие этапы:

* Составление справочной документации;
* Составление руководства пользователя;
* Составление руководства разработчика.

Это последний эта разработки. После завершения этого этапа реализовано готовое программное обеспечение. На рисунке 10 приведены последовательные этапы составления документации.

Рис 10. Составление документации программы «Журнал посещения занятий»

**ВЫВОДЫ**

В рамках выполнения данной курсовой работы написана программа «Журнал посещения занятий» в двух исполнениях: консольном и windows. Приведены краткая и полная схема разработки программы и дано описание модели проектирования журнала посещения занятий.

Разработка осуществлялась с использованием языка программирования Python 3. В обоих исполнениях программа считывает данные, введенные пользователем о посещении в текущем дне занятий, список которых определяется конфигурационными данными. По мере заполнения программа производит валидацию введенных данных и, если данные валидны, сохраняет их в журнал.

Программа протестирована ручным способом на работу в случаях ввода значений, отличающихся от валидных значений ввода, тесты пройдены успешно.

Так же в рамках выполнения курсовой работы приведены полная и краткая схема разработки программы. В рамках выполнения курсовой работы применялась каскадная модель разработки программного обеспечения.

В заключительном разделе описана модель проектирования программы. Приведены основные входные и выходные данные на каждом из этапов проектирования, а так же указаны входные компоненты для каждого этапа проектирования. Для каждого этапа приведено графическое описание с необходимым, в рамках выполнения курсовой работы, описанием.