Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

ITMO University

ОТЧЕТ ПО ДОМАШНЕМУ ЗАДАНИЮ

По дисциплине Программирование

Тема работы Домашнее задание

Обучающийся Зенин Данил Дмитриевич

Факультет Факультет инфокоммуникационных технологий

Группа К3120

Направление подготовки 11.03.02 Инфокоммуникационные технологии и системы связи

Образовательная программа Программирование в инфокоммуникационных системах

Обучающийся	_____ (дата)	_____ (подпись)	Зенин Д.Д. (Ф.И.О.)
Руководитель	_____ (дата)	_____ (подпись)	Казанова П.П. (Ф.И.О.)

СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ

В ходе работы будет создано консольное приложение для контроля собственных денежных средств, описаны этапы его создания и предоставлены блок-схемы работы алгоритмов.

Цель работы:

Создать программное обеспечение системы обработки данных: «Программа для контроля собственных денежных средств».

Задачи:

- Провести анализ предметной области и требований.
- Придумать алгоритмы для решения конкретных задач.
- Написать программу для данных алгоритмов на языке Python.
- Получить необходимый результат на выходе

1 Ход выполнения работы

Перед началом написания работы необходимо провести анализ предметной области и требований. Требуется создать простое приложение, которые помогут пользователям распределять собственный бюджет. Таким приложением может стать приложение для создания списка покупок. Подобные приложения чаще всего используется обычными пользователями, которые не обладают навыками программиста, поэтому необходимо создать программу, которая будет понятна и проста в использовании. Программа должно выполнять в полной мере функционал, необходимый пользователю.

Для выполнения задания была реализована программа, которая создает список покупок и позволяет пользователю взаимодействовать с ним.

Программа была написана в консольном приложении, используя вечный цикл(см. рисунок 1.1)

```
cart = []
while flag:
    flag0 = True
    while flag0:
        print("Возможные действия: \n1. Добавить продукт в корзину.\n", end='')
        print("2. Просматривать список продуктов.\n3. Просматривать покупки по дате и категории.\n", end='')
        print("4. Сортировка и вывод по возрастанию стоимости.\n",end='')
        print("5. Сортировка и вывод по убыванию стоимости.\n6. Удалить товар из списка.\n",end='')
        print("7. Выйти из программы")
        num = input("Введите номер действия: ").strip()
        if num.isdigit() and (num in ['1', '2', '3', '4', '5', '6', '7']):
            flag0 = False
        else:
            print("Введите корректный номер: ")
```

Рисунок 1.1 — Начало пользования приложением

Выводится сообщение, в котором просят ввести номер действия, которое пользователь хочет совершить. Корректность этого номера проверяется в еще одном вечном цикле с помощью метода isdigit() и проверки на вход в список возможных номеров(см. рисунок 1.1)

Для работы приложения были написаны некоторые функции(см. рисунок 1.2)

```
1
2   from calendar import isleap
3   import sys
4
5   print("Программа для контроля собственных денежных средств")
6   flag = True
7   def isFloat(num):
8       try:
9           float(num)
10      except (ValueError):
11          return False
12      return True
13
14
15  def sortByPrice(matrix):
16      for i in range(len(matrix)-1):
17          for j in range(len(matrix)-i-1):
18              if matrix[j][2] > matrix[j+1][2]:
19                  matrix[j], matrix[j+1] = matrix[j+1], matrix[j]
20      outputlst(matrix)
21
22
23  def sortByPriceReverse(matrix):
24      for i in range(len(matrix)-1):
25          for j in range(len(matrix)-i-1):
26              if matrix[j][2] < matrix[j+1][2]:
27                  matrix[j], matrix[j+1] = matrix[j+1], matrix[j]
28      outputlst(matrix)
```

Рисунок 1.2 — Функции сортировки и проверки на float

Функция isfloat(num) проверяет введенную строку на то, является ли она дробным числом с помощью конструкции try-except, которая выводит false, если встречает ошибку значения, и true в обратном случае. Она помогает избежать ошибок ввода(см. рисунок 1.2)

Функции sortByPrice(matrix) и sortByPriceReverse(matrix) сортируют массив по стоимости продукта с помощью пузырьковой сортировки(см. рисунок 1.2)

В вышеописанных функциях(см. рисунок 1.2) используется функция outputlst(matrix), которая выводит список покупок в консоль. Для лаконич-

ного вывода заголовков столбцов используем метод ljust, который делает длину не меньше введенного, добавляя пробелы в конце. Кроме того, с помощью счётчика count делаем номер для каждой строки прямо в выводе, делая по итогу маркированный список на выходе(см. рисунок 1.3)

```
31  def outputlst(matrix):
32      count = 1
33      print("-"*43+"Корзина продуктов"+"-"*43)
34      print('|'+'Имя'.ljust(35, ' ')+"|"+"Категория".ljust(21, ' ')+'|'
35            +"Стоимость".ljust(21, ' ')+'|'+"Дата".ljust(21, ' ')+'|')
36      for i in range(len(matrix)):
37          print('|'+str(count)+'. ',end='')
38          for j in range(4):
39              if j == 0:
40                  print((str(matrix[i][j]).ljust(32, ' ')+'|'), end='')
41              else:
42                  print((str(matrix[i][j]).ljust(21, ' ')+'|'), end='')
43          count+=1
44          print()
```

Рисунок 1.3 — Функция вывода матрицы

Функции searchByDate(matrix, date) и searchByCategory(matrix, date) (см. рисунок 1.4) выводят список только с теми покупками, дату или категорию которых укажет пользователь. В обоих случаях создается дополнительная матрица, куда записываются элементы основной матрицы, которые не равны введенной дате или категории. После идет проход по основной матрице, откуда удаляются элементы, если они равны элементам дополнительной матрицы, тем самым оставляя только те значения списка, у которых дата или категория такая, какую необходимо найти.

```
47  def searchByDate(matrix, date):
48      a = []
49      for i in range(len(matrix)):
50          if matrix[i][3].strip() != date:
51              a.append(i)
52      a.sort(reverse=True)
53      print(a)
54      for i in range(len(a)):
55          matrix.pop(a[i])
56      return matrix
57
58  def searchByCategory(matrix, category):
59      a = []
60      for i in range(len(matrix)):
61          if matrix[i][1].strip() != category:
62              a.append(i)
63      a.sort(reverse=True)
64      print(a)
65      for i in range(len(a)):
66          matrix.pop(a[i])
67      return matrix
```

Рисунок 1.4 — Функции поиска по входным данным

Всевозможные тесты на неверный ввод данных пользователем представлен в функции tests()(см. рисунок 1.5). Все проверки основаны на вечных циклах, которые просят пользователя ввести значение, пока не будет введено корректное. В случае со стоимостью продукта идет проверка на числовое значение ввода функцией isfloat(), далее эта стоимость округляется до 5 знаков после запятой, чтобы даже максимальная стоимость с дробной частью не ломала структуру вывода.

```
def tests():
    flag2 = True
    while flag2:
        price_of_product = input("Введите стоимость продукта(макс 999999999999999): ").strip()
        if isFloat(price_of_product):
            if float(price_of_product) >= 0 and float(price_of_product) <= 999999999999999:
                price_of_product = round(float(price_of_product), 5)
                flag2 = False
            else:
                print("Ввели неверную стоимость. Попробуйте снова.")
        else:
            print("Ввели неверную стоимость. Попробуйте снова.")
    flag3 = True
    while flag3:
        date_of_product = input("Введите дату реализации продукта(формат xx.xx.xxxx): ").strip()
        if (len(date_of_product) == 10 and date_of_product[:2].isdigit() and date_of_product[3:5].isdigit()
        and date_of_product[6:].isdigit()):
            if (1 <= int(date_of_product[:2]) <= 31 and 1 <= int(date_of_product[3:5]) <= 12
                and 1900 <= int(date_of_product[6:]) <= 2023
                and checkDate(int(date_of_product[:2]), int(date_of_product[3:5]), int(date_of_product[6:]))):
                flag3 = False
            else:
                print("Вы ввели неверную дату. Попробуйте снова")
        else:
            print("Вы ввели неверную дату. Попробуйте снова")
    return (price_of_product, date_of_product)
```

Рисунок 1.5 — Функция тестов входных данных

Алгоритм проверки введенных данных пользователем представлен на блок-схеме(см. рисунок 1.6):

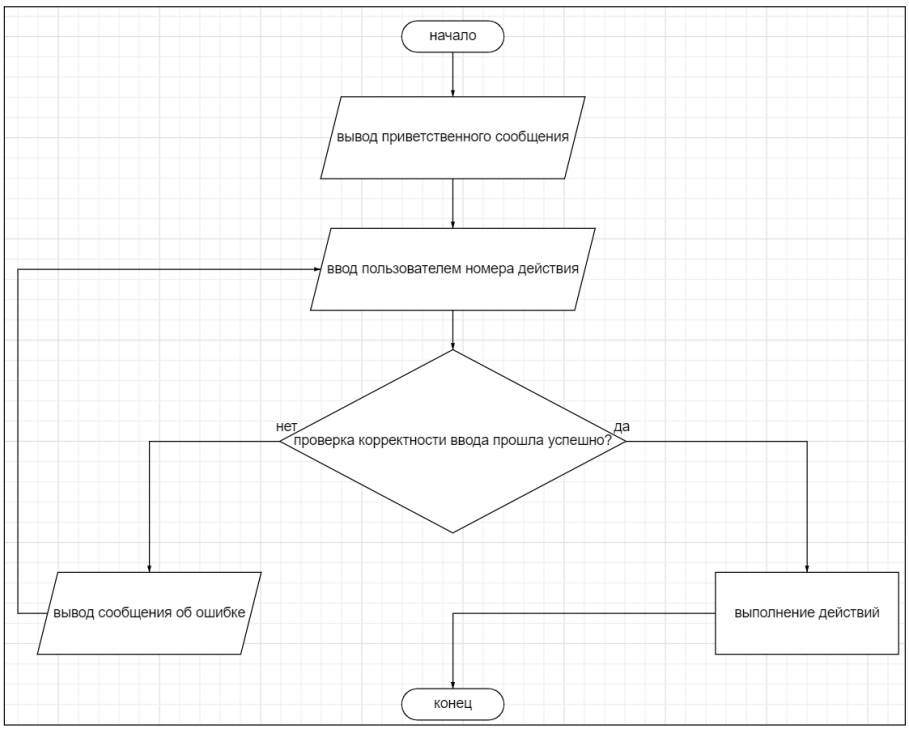

Рисунок 1.6 — Блок-схема алгоритма проверки входных данных

Также в функции tests() (см. рисунок 1.5) использовалась функция checkDate(day, month, year)(см. рисунок 1.7). Создается матрица с кортежами из номера месяца и количества дней в нем. Ячейка с февралем является списком, чтобы далее при проверке года на високосность можно было поменять количество дней в феврале на 29. После введенные данные проверяются на соответствие с нужным элементом матрицы.

```
97  def checkDate(day, month, year):
98      a = [(1, 31), [2, 28], (3, 31), (4, 30), (5, 31), (6, 30), (7,31), (8, 31), (9, 30), (10, 31),
99          (11, 30), (12, 30)]
100     if month == 2:
101         if isleap(year):
102             a[1][1] = 29
103     if a[month-1][1] >= day:
104         return True
```

Рисунок 1.7 — Функция проверки даты

Система реагирует на ввод пользователем цифры 1(см. рисунок 1.8). Инициализируются входные данные и заносятся в список, который заносится в матрицу.

```
123     if num == '1':
124         name_of_product = input("Введите имя продукта: ").strip().lower()
125         category_of_product = input("Введите категорию продукта: ").strip().lower()
126         result = tests()
127         price_of_product, date_of_product = result[0], result[1]
128         ls = [name_of_product, category_of_product, price_of_product, date_of_product]
129         cart.append(ls)
130         print("Продукт добавлен в корзину.")
```

Рисунок 1.8 — Реакция программы на "1"

Система реагирует на ввод пользователем цифры 2 (см. рисунок 1.9), выводя матрицу с помощью функции outputlst()

```
131     elif num == '2':
132         outputlst(cart)
```

Рисунок 1.9 — Реакция программы на "2"

Система реагирует на ввод пользователем цифры 3(см. рисунок 1.10). Выводится сообщение с вариантами действий для пользователя и входные данные обрабатываются в вечном цикле, чтобы избежать ввод некорректных данных. При вводе "1"идет проверка на правильность вводимых данных. При вводе "1"или "2"создается копия матрицы, чтобы исходная матрица не изменялась при просмотре по дате или категории. Копию заносится в функцию searchByDate или searchByCategory и выводится результат.

```
133     elif num == "3":
134         flag4 = True
135         while flag4:
136             com = input("Что вы хотите сделать? Введите команду. \n 1. Просмотр по дате."+
137                         "\n 2. Просмотр по категории\n").strip()
138             if com in ['1', '2']:
139                 flag4 = False
140             else:
141                 print("Неверная команда. Попробуйте ещё раз.")
142         flag4 = True
143         if com == '1':
144             while flag4:
145                 inp = input("Введите дату реализации продукта(формат xx.xx.xxxx): ").strip()
146                 if len(inp) == 10 and inp[:2].isdigit() and inp[3:5].isdigit() and inp[6:].isdigit():
147                     if (1 <= int(inp[:2]) <= 31 and 1 <= int(inp[3:5]) <= 31 and 1900 <= int(inp[6:]) <= 2023
148                     and checkDate(int(inp[:2]), int(inp[3:5]), int(inp[6:]))):
149                         flag4 = False
150                     else:
151                         print("Вы ввели неверную дату. Попробуйте снова")
152                 else:
153                     print("Вы ввели неверную дату. Попробуйте снова")
154             lst = cart.copy()
155             z = searchByDate(lst, inp)
156             print("Список продуктов по дате")
157             outputlst(z)
158         if com == '2':
159             category = input("Введите категорию товара: ").strip().lower()
160             lis = cart.copy()
161             x = searchByCategory(lis, category)
162             outputlst(x)
```

Рисунок 1.10 — Реакция программы на "3"

Система реагирует на ввод пользователем цифры 4(см. рисунок 1.11),сортируя матрицу с помощью функции sortByPrice()

```
164     elif num == '4':
165         sortByPrice(cart)
```

Рисунок 1.11 — Реакция программы на "4"

Система реагирует на ввод пользователем цифры 5 (см. рисунок 1.12), обратно сортируя матрицу с помощью функции sortByPriceReverse()

```
166     elif num == '5':
167         sortByPriceReverse(cart)
```

Рисунок 1.12 — Реакция программы на "5"

Система реагирует на ввод пользователем цифры 6(см. рисунок 1.13). Генератором списков создается список возможных номеров в корзине. Далее

идет ввод пользователем номера продукта, который он хочет удалить, и проверка правильности ввода вечным циклом

```
168     elif num == '6':
169         a = [str(i) for i in range(1, len(cart)+1)]
170         flag1 = True
171         while flag1:
172             result = input("Введите номер продукта в корзине, который хотите удалить: ").strip()
173             if (result in a):
174                 cart.pop(int(result)-1)
175                 break
176             else:
177                 print("Введите корректный номер продукта в таблице")
178 
179         outputlst(cart)
```

Рисунок 1.13 — Реакция программы на "6"

Система реагирует на ввод пользователем цифры 7(см. рисунок 1.14), меняя значение flag не False, останавливая главный вечный цикл и завершая программу.:

```
180     elif num == '7':
181         flag = False
182 
183 
184 sys.stdout = open("text.txt", "w")
185 outputlst(cart)
```

Рисунок 1.14 — Реакция программы на "7"

После ввода цифры 7 результат сохраняется в заранее созданный для этого текстовый файл.

Общий алгоритм работы программы представлен на блок-схеме(см. рисунок 1.15):

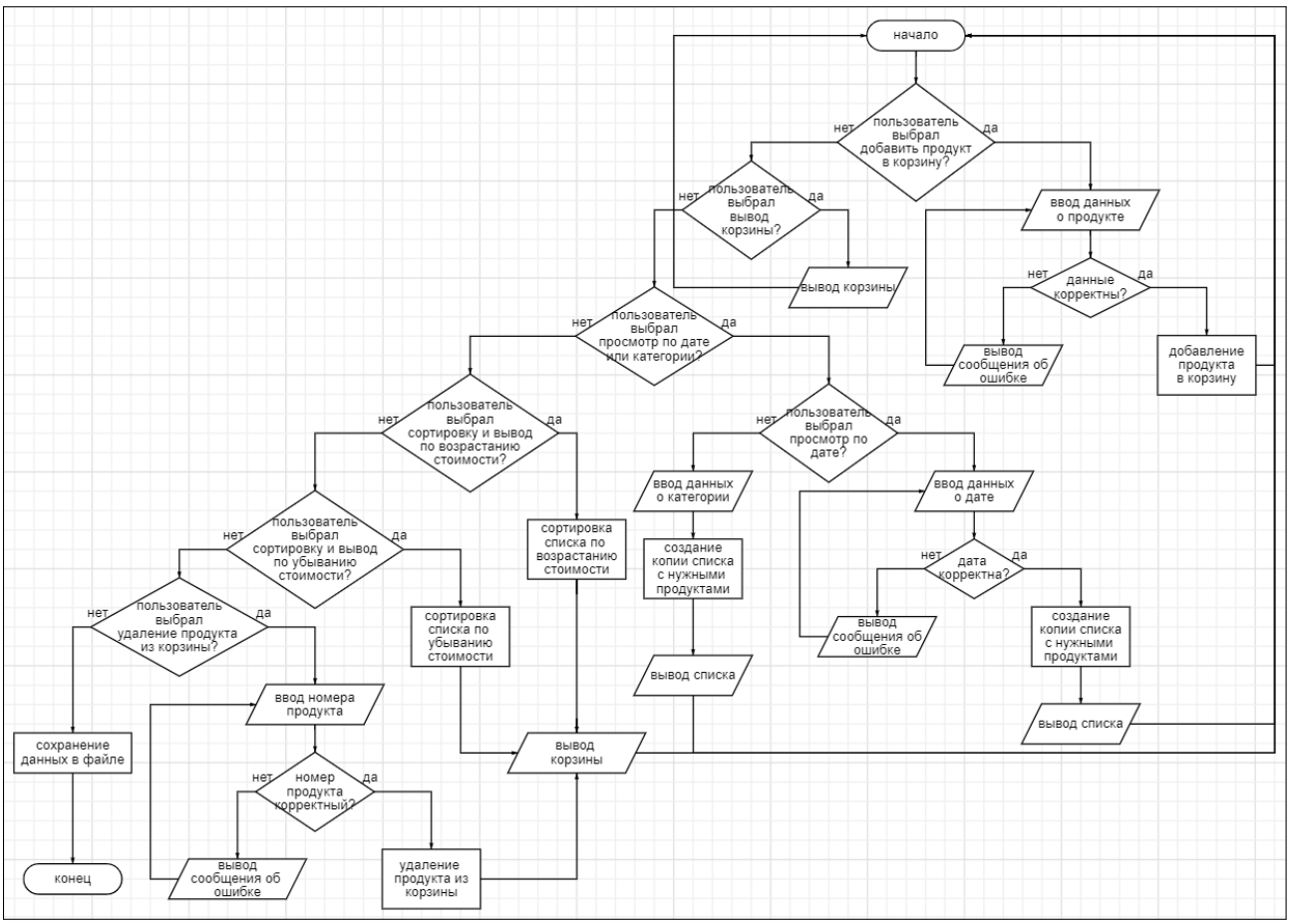

Рисунок 1.15 — Алгоритм работы программы

ВЫВОДЫ

В ходе работы было создано консольное приложение для контроля собственных денежных средств, описаны этапы его создания и предоставлены блок-схемы работы алгоритмов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. UML app.diagrams URL: [Электронный ресурс]: [сайт].https://app.diagrams.net/ (Дата обращения 30.10.2023)

2. Код программы на GitHUb URL: https://github.com/dellup/aisd1/blob/main/proga.py