IV Международный конкурс
исследовательских работ школьников
«Research start 2021/2022»

Государственное бюджетное образовательное учреждение
лицей-интернат «Центр одаренных детей»

Проектная работа

СОЗДАНИЕ КАЛЬКУЛЯТОРА ТРОИЧНОЙ УРАВНОВЕШЕННОЙ СИСТЕМЫ СЧИСЛЕНИЯ В СРЕДЕ PYTHON

Работу выполнил: Прохин Михаил,
учащийся 10Г класса

Руководитель: Кузина Ольга Владимировна,
учитель информатики

Нижний Новгород
2022

Содержание

Введение

В ходе изучения курса информатики на уроке по системам счисления, отличным от двоичной, меня заинтересовала троичная уравновешенная система счисления (ТУСС) и связанная с ней троичная логика, которая легла в основу малой ЭВМ «Сетунь», созданной в МГУ ещё в 1959 году.

Я решил подробнее изучить эту систему счисления, алгоритмы перевода чисел между десятичной системой счисления и ТУСС, осуществление сложения (вычитания), умножения и деления в ней. При работе с различными системами счисления я нередко подстраховываюсь с помощью калькулятора, которого, к сожалению, для ТУСС найти не смог, что стало причиной выбора темы для данного проекта.

Было необходимо выбрать среду для реализации проекта. В итоге был избран Python. Синтаксис этого языка прост в изучении, с ним я знаком, так как в лицее он является основным используемым при программировании на уроках информатики. Интерпретатор Python бесплатен, как и доступ к библиотекам, например PyQt, с помощью которой будет реализовано графическое оформление программы.

Объекты исследования: троичная уравновешенная система счисления и язык программирования Python.

Предмет исследования: алгоритмы выполнения арифметических действий в ТУСС и их реализация в Python.

Цель проекта: создание калькулятора, работающего с ТУСС, используя среду Python.

Задачи проекта:

1. Изучить алгоритм перевода чисел в ТУСС.
2. Изучить осуществление арифметических операций в ТУСС.
3. Представить данные алгоритмы в Python.
4. Создать приложение-калькулятор троичной уравновешенной системы счисления в среде Python.

Троичная уравновешенная система счисления

Троичная уравновешенная система счисления – система счисления с основанием **3** и алфавитом {-1, 0, 1}. Почему же она названа уравновешенной? Чтобы ответить на этот вопрос и понять её отличие от двоичной СС, предлагаю решить две задачи:

1. Найдите такой набор из **5 гирек**, чтобы, кладя их на одну чашу весов, можно было бы взвесить груз массой от 1 до 31 унции ($m \in \mathbb{N}$).
2. Найдите такой набор из **4 гирек**, что с их помощью на равноплечих весах можно было бы взвесить груз массой от 1 до 40 унций ($m \in \mathbb{N}$).

Ответом на первую задачу является набор из гирек, массы которых: 1, 2, 4, 8, 16 унций. В этом случае для каждой гирьки мы решаем, кладём ли мы её (1) или нет (0), взвешивая груз от 1 (00001_2) до 31 (11111_2) унции.

Ответом на вторую является набор из гирек, массы которых: 1, 3, 9, 27 унций. Здесь же для каждой гири мы решаем, кладём ли мы её в противовес грузу (1), не кладём (0) или кладём к грузу (-1), **уравновешивая** весы для нахождения массы груза от 1 ($00001_{3ур}$) до 40 ($11111_{3ур}$) унций. Так и работает ТУСС, в которой числа представляются в виде суммы/разницы степеней тройки. Заметим, что, используя меньше гирек, во второй задаче можно взвесить больше различных по массе грузов.

Так будут представлены некоторые числа в троичной уравновешенной системы счисления (для удобства записи чисел в виде $N_{3ур}$ введём символ $\bar{1}$):

$62 = (81 - 27 + 9 - 1)_{10} = 1\bar{1}10\bar{1}_{3ур}$

$168 = (243 - 81 + 9 - 3)_{10} = 1\bar{1}01\bar{1}0_{3ур}$

$75 = (81 - 9 + 3)_{10} = 10\bar{1}10_{3ур}$

Но гадать, какие степени тройки будут добавлены или вычтены, не удобно. Поэтому необходим алгоритм перевода из десятичной системы счисления в троичную уравновешенную.

Алгоритм перевода чисел в ТУСС

1) Перевести модуль числа в троичную систему счисления
2) Двойки заменить на $\bar{1}$, а к следующему разряду добавить 1
3) Если число было отрицательным, то заменить 1 на $\bar{1}$, $\bar{1}$ на 1

Основные преимущества троичной симметричной системы счисления (перед двоичной)

- естественное представление положительных и отрицательных чисел позволяет не использовать дополнительный код числа
- знак числа задается его наиболее значимым ненулевым тритом (если старший разряд 1, то число положительно, $\bar{1}$ – отрицательно)
- отрицание числа осуществляется взаимной заменой 1 и -1 ($-8_{10}= \bar{1}01_{Зур}$; $8_{10}= 10\bar{1}_{Зур}$)
- для представления чисел необходимо меньше (не больше) символов

Арифметические операции в ТУСС

Основой операцией, является сложение. Начиная с конца, в зависимости от суммы соответствующих разрядов первого, второго слагаемых и переполнения из предыдущего разряда (изначально равного нулю) в начало результата будут дописываться соответствующее значение, и меняться переполнение. В конце старшим разрядом станет последнее переполнение, незначащие нули отбрасываются.

Полностью таблица сложения, которую можно использовать для обычного сложения чисел столбиком, имеет вид:

1̄	**+**	**1̄**	**0**	**1**	**0**	**+**	**1̄**	**0**	**1**	**1**	**+**	**1̄**	**0**	**1**
	1̄	1̄0	1̄1	1̄		**1̄**	1̄1	Ī	0		**1̄**	1̄	0	1
	0	1̄1	1̄	0		**0**	Ī	0	1		**0**	0	1	11̄
	1	1̄	0	1		**1**	0	1	11̄		**1**	1	11̄	10

Сложим в уравновешенной троичной системе числа 412 и 181:

$412_{10} = 120021_3 = 1\bar{1}\bar{1}01\bar{1}1_{Зур}$.; $181_{10} = 20201_3 = 1\bar{1}1\bar{1}01_{Зур}$.

$1\bar{1}1100\bar{1}_3 = 593_{10} = 412 + 181$ верно.

1	1̄	1̄	0	1	1̄	1
+	1	1̄	1	1̄	0	1
1	1̄	1	1	0	0	1̄

Пользуясь тем, что вычитание – сложение с противоположным второму слагаемому, его можно осуществить через сложение с числом, в котором взаимно поменяны 1 и 1̄.

Умножение чисел сводится к простым операциям изменения знака (при необходимости) и сложения. Умножим, например, 5_{10} на 12_{10}:

$5_{10} = 1\bar{1}\bar{1}_3$

$12_{10} = 110_3$

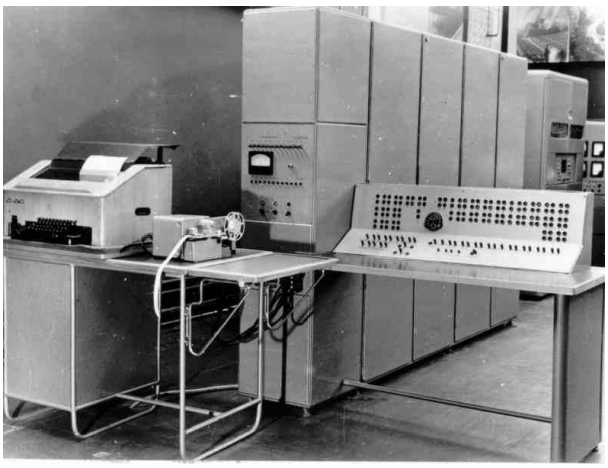
		1	1̄	1̄
	*	1	1	0
	1	1̄	1̄	
1	1̄	1̄		
1	1̄	1	1̄	0

Переведя троичное число $1\bar{1}1\bar{1}0$ в десятичную систему, получим 60.

ЭВМ «Сетунь»

Уравновешенная троичная система счисления применялась в ЭВМ «Сетунь», разработанной в 1959 году в МГУ им. М.В. Ломоносова под руководством Николая Петровича Брусенцова (на фото).

Минимальной адресуемой единицей памяти «Сетуни» стал трайт, равный шести тритам и принимающий значения от -364 до 364. Работа с диапазоном отрицательных значений - особенность, отличающая трайт от двоичного байта, значения которого распространяются от 0 до 255. С помощью трайта можно закодировать все заглавные и строчные символы русского и латинского алфавитов, необходимые математические и служебные символы.

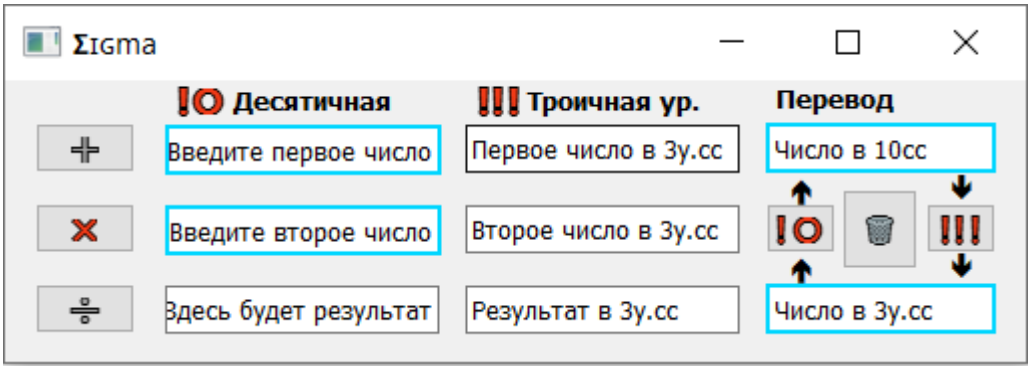


На фото - промышленный образец ЭВМ «Сетунь», ВДНХ, 1961 г. (Сетунь — название речки, протекающей неподалеку от МГУ.)

Сегодня IBM, Motorola, Hypres и TexasInstruments ведут различные исследования в области троичных компьютеров.

Создание калькулятора троичной уравновешенной системы счисления на языке программирования Python

Для создания программ с графическим интерфейсом на языке Python существует большое количество библиотек. Я воспользовался библиотекой PyQt. Изучив ранее описанные алгоритмы и пройдя на образовательной платформе Stepik курс «Поколение Python: курс для продвинутых», предложенный учителем, я приступил к написанию программ, реализующих перевод чисел из десятичной системы счисления в ТУСС и обратно, сложения (вычитания) и умножения в ТУСС. В качестве эксперимента была добавлена функция целочисленного деления (div или //), которую в ходе дальнейшего совершенствования планируется исправить. Ниже представлен интерфейс приложения.



Далее я приступил к «наполнению» приложения. Окно для удобства разделено на несколько частей: кнопки, ввод чисел, вывод их записей в ТУСС, перевод между десятичной и троичной уравновешенной системами счисления. В последней также вы можете заметить кнопку для очищения всех полей ввода/вывода. При нажатии кнопок арифметических операций или перевода, если ввод был некорректен, в результате выведет надпись об этом.

Код созданного приложения с комментариями представлен в приложении:

Заключение

В ходе работы над проектом была исследована троичная уравновешенная система счисления и алгоритмы работы в ней.

Средствами языка программирования Python были реализованы следующие алгоритмы:

- Перевод чисел из десятичной системы счисления в ТУСС.
- Перевод чисел из ТУСС в десятичную.
- Арифметические операции (сложение (вычитание), умножение, деление) в ТУСС.

С помощью графической библиотеки PyQt было создано оконное приложение, позволяющее с числами в ТУСС.

Данное приложение может быть использовано на уроках информатики при изучении систем счисления.

Литература и источники информации

1. Троичная арифметика (trinary.su)
2. Прохоренок Н.А. PyQt. Создание оконных приложений на Python. Электронное авторское издание, 2011.
3. Сетунь (компьютер) — Википедия (wikipedia.org)

Приложение

```
import sys
from PyQt5 import uic
from PyQt5.QtWidgets import QApplication, QWidget, QMainWindow, QLabel
from PyQt5.QtWidgets import QPushButton, QLineEdit, QInputDialog, QStyle
def dec(zap):
    d = {'i': -1, '0': 0, '1': 1}
    s = [d[zap[i]] * 3**(len(zap) - i - 1) for i in range(len(zap))]
    return sum(s)
def SUM(S, d):                  # Ŧ
    per = {-3: ('i', '0'), -2: ('i', '1'), -1: ('0', 'i'), 0: ('0', '0'),
           1: ('0', '1'), 2: ('1', 'i'), 3: ('1', '0')}
    p = '0'
    Ss = ''
    if len(S) > len(d):
        d = '0' * (len(S) - len(d)) + d
    elif len(d) > len(S):
        S = '0' * (len(d) - len(S)) + S
    for i in range(1, len(S) + 1):
        p, s = per[dec(S[-i]) + dec(d[-i]) + dec(p)]
        Ss = s + Ss
    Ss = p + Ss
    while len(Ss) > 1 and Ss[0] == '0':
        Ss = Ss[1:]
    return Ss
def check(Text):
    if Text and (Text.isdigit() or Text[0] == '-' and Text.replace('-','',1).isdigit()):
        return True
    elif Text and (Text[0] == '+' and Text.replace('+','',1).isdigit()):
        return True
    else:
        return False
def tri(Dec):
    S = '' #ОсНоВа
    Mod = abs(Dec)
    while Mod > 0:
        S = str(Mod % 3) + S
        Mod = Mod // 3
    d = S.replace('1','0').replace('2','1') + '0'
    S = S.replace('2','i')
    Ss = SUM(S, d)
    if Dec < 0:
```

```
        Ss = Ss.replace('1','z').replace('i','1').replace('z','i')
    return Ss
StyleSheet = '''
    QLineEdit {
        border-width: 2px;
        border-style: solid;
        border-color: rgb(0, 214, 255);
    }
    QLineEdit:focus {
        border-color: rgb(255, 156, 0);
    }
'''

class Example(QWidget):
    def __init__(self):
        super().__init__()
        self.initUI()
    def initUI(self):                    #_Геометрия_кнопок_и_ввода/вывода_
        self.btn1 = QPushButton('+', self)   # Кнопка сложения
        self.btn1.resize(50, 25)
        self.btn1.move(15, 21)
        self.le1 = QLineEdit(self)           # Ввод первого числа
        self.le1.move(80, 22)
        self.le1.setStyleSheet(StyleSheet)
        self.le1.resize(138, 25)
        self.le13 = QLineEdit(self)          # Вывод 3 ур. записи первого
        self.le13.move(230, 22)
        self.btn1.clicked.connect(self.key1)
        self.btn2 = QPushButton('×', self)    # Кнопка умножения
        self.btn2.resize(50, 25)
        self.btn2.move(15, 61)
        self.le2 = QLineEdit(self)           # Ввод второго числа
        self.le2.move(80, 62)
        self.le2.setStyleSheet(StyleSheet)
        self.le2.resize(138, 25)
        self.le23 = QLineEdit(self)          # Вывод 3 ур. записи второго
        self.le23.move(230, 62)
        self.btn2.clicked.connect(self.key2)
        self.btn3 = QPushButton('÷', self)    # Кнопка цел. деления
        self.btn3.resize(50, 25)
        self.btn3.move(15, 101)
```

```
self.le3 = QLineEdit(self)            # Вывод результата в 10й
self.le3.move(80, 102)
self.le33 = QLineEdit(self)           # Вывод результата в 3й ур.
self.le33.move(230, 102)
self.btn3.clicked.connect(self.key3)
self.btnP10 = QPushButton('1 0', self) # Кнопка перевода 3 => 10
self.btnP10.resize(35, 25)
self.btnP10.move(380, 61)
self.leP10 = QLineEdit(self)          # Ввод числа 3 => 10
self.leP10.move(380, 21)
self.leP10.setStyleSheet(StyleSheet)
self.leP10.resize(115, 25)
self.leP3 = QLineEdit(self)           # Ввод числа 10 => 3
self.leP3.move(380, 101)
self.leP3.setStyleSheet(StyleSheet)
self.leP3.resize(115, 25)
self.btnP3 = QPushButton('1 1 1', self)  # Кнопка перевода 10 => 3
self.btnP3.resize(35, 25)
self.btnP3.move(460, 61)
self.btnP3.clicked.connect(self.Perevod3)
self.btnP10.clicked.connect(self.Perevod10)
self.clear = QPushButton('🗑', self)    # Кнопка очистки надписей
self.clear.resize(38, 40)
self.clear.move(418, 54)
self.clear.clicked.connect(self.Chistka) #_____________________
self.lbl1 = QLabel('1 0 Десятичная', self)  # Подписи
self.lbl1.move(85, 1)
self.lbl1.setStyleSheet("'font:bold'")
self.lbl2 = QLabel('1 1 1 Троичная ур.', self)
self.lbl2.setStyleSheet("'font:bold'")
self.lbl2.move(235, 1)
self.lbl3 = QLabel('Перевод', self)
self.lbl3.move(385, 1)
self.lbl3.setStyleSheet("'font:bold'")
self.lbl4 = QLabel('□              □', self)
self.lbl4.move(392, 45)
self.lbl4 = QLabel('□              □', self)
self.lbl4.move(392, 84)

self.le1.setText('Введите первое число')  # Пояснения
self.le13.setText('Первое число в 3у.сс')
```

```
        self.le2.setText('Введите второе число')
        self.le23.setText('Второе число в 3у.сс')
        self.le3.setText('Здесь будет результат')
        self.le33.setText('Результат в 3у.сс')
        self.leP10.setText('Число в 10сс')
        self.leP3.setText('Число в 3у.сс')

        self.setGeometry(670, 300, 510, 140)
        self.setWindowTitle('Σιgma')
        self.show()

    def key1(self):            # СЛОЖЕНИЕ
        if check(self.le1.text()) and check(self.le2.text()):
            Dec1 = int(self.le1.text())
            Dec2 = int(self.le2.text())
        else:
            self.le3.setText('Error')
            if self.le2.text() and self.le1.text():
                self.le33.setText('Некорректный ввод')
            else:
                self.le33.setText('Введите ДВА числа')
            return 0

        Ss1 = tri(Dec1)
        self.le13.setText(str(Ss1))
        Ss2 = tri(Dec2)
        self.le23.setText(str(Ss2))
        Ssi = SUM(Ss1, Ss2)
        self.le33.setText(Ssi)
        self.le3.setText(str(dec(Ssi)))

    def key2(self):                #УМНОЖЕНИЕ
        if check(self.le1.text()) and check(self.le2.text()):
            Dec1 = int(self.le1.text())
            Dec2 = int(self.le2.text())
        else:
            self.le3.setText('Error')
            self.le33.setText('Некорректный ввод')
            return 0

        Ss1 = tri(Dec1)
        self.le13.setText(str(Ss1))
```

```
    Ss2 = tri(Dec2)
    self.le23.setText(str(Ss2))

    Ssi = '0'
    for m in range(len(Ss2)):
        if Ss2[-m-1] == '1':
            Ssi = SUM(Ssi, Ss1 + m * '0')
        elif Ss2[-m-1] == 'i':
            Ssi = SUM(Ssi, Ss1.replace('1','z').replace('i','1').replace('z','i') + m * '0')
    self.le33.setText(Ssi)
    self.le3.setText(str(dec(Ssi)))

def key3(self):                #ЦЕЛОЧИСЛЕННОЕ ДЕЛЕНИЕ
    if check(self.le1.text()) and check(self.le2.text()):
        Dec1 = int(self.le1.text())
        Dec2 = int(self.le2.text())
    else:
        self.le3.setText('Error')
        self.le33.setText('Некорректный ввод')
        return 0

    Ss1 = tri(Dec1)
    self.le13.setText(str(Ss1))
    Ss2 = tri(Dec2)
    self.le23.setText(str(Ss2))

    Ssi, Snos = '', Ss1
    vag = Snos[:len(Ss2)]
    Snos = Snos[len(Ss2):]
    if abs(dec(vag)) < abs(dec(Ss2)):
        vag += Snos[0]
        Snos = Snos[1:]
    for i in range(len(Snos)+1):
        vag = tri(dec(vag))
        if len(vag) < len(Ss2):
            Ssi += '0'
        elif vag[0] == Ss2[0] != '0':
            Ssi += '1'
            vag = SUM(vag, Ss2.replace('1','z').replace('i','1').replace('z','i'))
        elif vag[0] != Ss2[0] != '0' and vag[0] != '0':
            Ssi += 'i'
            vag = SUM(vag, Ss2)
```

```
            if Snos:
                vag += Snos[0]
                Snos = Snos[1:]
        if len(vag) == len(Ss2):
            if vag[0] == Ss2[0] != '0':
                Ssi = SUM(Ssi, '1')
            if vag[0] != Ss2[0] != '0' and vag[0] != '0':
                Ssi = SUM(Ssi, 'i')

        self.le33.setText(Ssi)
        self.le3.setText(str(dec(Ssi)))

        if Ss2 == '0':
            self.le3.setText('Error')
            self.le33.setText('')
    def Perevod3(self):                # ПЕРЕВОД ИЗ 10 в 3ур
        if self.leP10.text().isdigit():
            self.leP3.setText(tri(int(self.leP10.text())))
        else:
            self.leP3.setText('Не число 10сс')
    def Perevod10(self):               # ПЕРЕВОД ИЗ 3ур в 10
        if set(self.leP3.text()) | {'0', '1', 'i'} == {'0', '1', 'i'}:
            self.leP10.setText(str(dec(self.leP3.text())))
        else:
            self.leP10.setText('В 3у.сс только i01')
    def Chistka(self):                 # ОЧИСТКА ПОЛЕЙ ВВОДА
        self.le1.setText('')
        self.le2.setText('')
        self.le3.setText('')
        self.le13.setText('')
        self.le23.setText('')
        self.le33.setText('')
        self.leP10.setText('')
        self.leP3.setText('')
if __name__ == '__main__':
    app = QApplication(sys.argv)
    ex = Example()

    sys.exit(app.exec_())
```