Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Кафедра «Вычислительная техника»

Дисциплина «Высокопроизводительные вычисления»

**Лабораторная работа № 2**

**Вариант 10**

**Реализация генератора псевдослучайных чисел**

Выполнил:

Студент группы ИВТАСбд-42

Сулейманов М.З.

Проверил:

Мартынов А.И.

Ульяновск

2024

**Цель работы**

Научиться генерировать псевдослучайную последовательность с помощью

Генератора Парка-Миллера.

Задание

Реализовать приложение интерфейсом, позволяющее выполнять следующие действия:

1. Генерировать последовательность в битах (длина последовательности задается) с помощью выбранного генератора

2. Сохранять полученную последовательность в файл и выводить ее на экран приложения

3. Загружать последовательность из файла

4. Проверять полученную последовательность с помощью тестов, реализованных в предыдущей лабораторной работе. Результат проверки должен отображаться в приложении

**Генератор Парка-Миллера**

Линейными конгруэнтными генераторами являются генераторы следующей формы:



в которых X n –n-й член последовательности, а X n−1 – предыдущий член последовательности. Переменные a, b и m – постоянные: а – множитель, b – инкремент и m – модуль. Ключом или затравкой служит значение X0 . Период такого генератора не больше, чем m. Если a, b и m подобраны правильно, то генератор будет генератором с максимальным периодом, и его период будет равен m. (Например, для линейного конгруэнтного генератора b должно быть взаимно простым с m). Если инкремент b равен нулю, то есть генератор имеет вид:



и мы получим самую простую последовательность, которую можно предложить для генератора с равномерным распределением. При соответствующем выборе констант a = 7 5 = 16 807 и m = 2 31 -1 = 2 147 483 647 мы получим генератор с максимальным периодом повторения. Эти константы были предложены учеными Парком и Миллером, поэтому генератор вида:



называется генератором Парка-Миллера.

**Листинг**

|  |
| --- |
| from tkinter import \*  from tkinter import Tk, RIGHT, BOTH, RAISED  from tkinter.ttk import Frame, Button, Style  from textwrap import wrap  from tkinter import ttk  import math  global s #глобальная переменная которая будет содержать последовательность символов  s="1" #Просто чтобы пустой не была вдруг поломается что-то  sizeof=0  def park\_miller(seed, a=16807, m=2147483647):      """      Генератор псевдослучайных чисел Парка-Миллера.      Args:          seed: Целое число, начальное значение (seed).  Должно быть положительным.          a: Мультипликатор (по умолчанию 16807).          m: Модуль (по умолчанию 2147483647).      Yields:          Последовательность псевдослучайных чисел в диапазоне [0, 1).      """      if seed <= 0:          raise ValueError("Приближение должно быть положительным")      x = seed      while True:          x = (a \* x) % m          random\_number = x / m          if random\_number < 0.5:              yield 0          else:              yield 1  def load(): #Функция для кнопки загрузки последовательности из файла      global s #Вызов глобальной перменной      lbl.delete("1.0", END) # Очищаем Text виджет      lbl.insert(END, "Загружено") # Вставляем текст в Text виджет      with open("file.txt", "r") as input: #Открытие файла          s=input.read() #Чтение      lbl.delete("1.0", END) # Очищаем Text виджет      lbl.insert(END, s) # Вставляем текст в Text виджет  def check(): #Функция для кнопки проверки загруженной/сгенерированной последовательности      global s #Вызов глобальной перменной      sum=0 #Будущая сумма чисел последовательности      for i in range(len(s)): #Получение суммы чисел последовательности          if(s[i])=='1':              sum+=1          else:              sum-=1      if((abs(sum)/math.sqrt(len(s)))<1.82138636): #Проверка последовательности          lbl.delete("1.0", END) # Очищаем Text виджет          lbl.insert(END, "Частотный тест пройден успешно!") # Вставляем текст в Text виджет      else:          lbl.delete("1.0", END) # Очищаем Text виджет          lbl.insert(END, "Последовательность не прошла частотный тест") #  def generate(): #Функция для кнопки генерации последовательности      global s #Вызов глобальной перменной      seed=int(seedEntry.get())      sizeof=int(sizeEntry.get()) #получение размера последовательности из поля для ввода      if(sizeof!=0):          generator = park\_miller(seed)          random\_numbers=[next(generator) for \_ in range(sizeof)]          for i in range(sizeof):              random\_numbers[i]=str(random\_numbers[i])          s=''.join(random\_numbers)          lbl.delete("1.0", END) # Очищаем Text виджет          lbl.insert(END, s) # Вставляем текст в Text виджет          with open("file.txt", "w") as output: #Открытие файла              output.write(s) #Запись      else:          lbl.delete("1.0", END) # Очищаем Text виджет          lbl.insert(END, "Введите значение в поле справа от кнопки Сгенерировать") # Вставляем текст в Text виджет  #Создание окна и кнопок  window = Tk()  window.title("Частотный тест")  window.geometry('800x620')  lbl = Text(window, font=("Arial", 10), wrap=WORD) # wrap=WORD для переноса слов  lbl.pack(fill=BOTH, expand=True)  scrollbar = ttk.Scrollbar(window, orient="vertical", command=lbl.yview)  scrollbar.pack(side=RIGHT, fill=Y)  lbl.config(yscrollcommand=scrollbar.set) # Связываем Text с полосой прокрутки  generateButton = Button(text="Сгенерировать", command=generate)  generateButton.pack(side=RIGHT, padx=5, pady=5)  loadButton = Button(text="Загрузить", command=load)  loadButton.pack(side=RIGHT)  generateButton = Button(text="Проверить", command=check)  generateButton.pack(side=RIGHT, padx=5, pady=5)  def on\_focus\_in\_size(event):      if event.widget.get() == "Количество бит":          event.widget.delete(0, END)          event.widget.config(fg="black")  def on\_focus\_out\_size(event):      if event.widget.get() == "":          event.widget.insert(0, "Количество бит")          event.widget.config(fg="gray")  sizeEntry=Entry(font=("Arial",10), text="asdasd")  sizeEntry.pack(side=RIGHT)  sizeEntry.insert(0,"Количество бит")  sizeEntry.config(fg="gray")  sizeEntry.bind("<FocusIn>", on\_focus\_in\_size)  sizeEntry.bind("<FocusOut>", on\_focus\_out\_size)  def on\_focus\_in\_seed(event):      if event.widget.get() == "Начальное приближение":          event.widget.delete(0, END)          event.widget.config(fg="black")  def on\_focus\_out\_seed(event):      if event.widget.get() == "":          event.widget.insert(0, "Начальное приближение")          event.widget.config(fg="gray")  seedEntry = Entry(font=("Arial", 10))  seedEntry.pack(side=RIGHT)  seedEntry.insert(0, "Начальное приближение")  seedEntry.config(fg="gray")  seedEntry.bind("<FocusIn>", on\_focus\_in\_seed)  seedEntry.bind("<FocusOut>", on\_focus\_out\_seed)  window.mainloop() |