Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

ТЕОРИЯ ПСЕВДОСЛУЧАЙНЫХ ГЕНЕРАТОРОВ

ОТЧЕТ ПО ПРАКТИЧЕСКОМУ КУРСУ

студента 4 курса 431 группы

факультета компьютерных наук и информационных технологий

Черватюка Сергея Александровича

фамилия, имя, отчество

Научный руководитель

Ст. преподаватель _____ И.И. Слеповичев

подпись, дата

Саратов 2024

Задание 2. Преобразование ПСЧ к заданному распределению

Необходимо создать программу, преобразовывающую ПСЧ в другую последовательность ПСЧ с заданным распределением. Входные параметры для алгоритмов передаются с помощью файла, название которого указывается в строке параметров запуска программы (p1, p2, p3). Преобразованные числа записываются в файл distr-xx.dat, где <xx> – код распределения. На вход в качестве последовательности ПСЧ подаётся последовательность, сгенерированная с помощью аддитивного генератора с входными параметрами: /g:add /n:10000 /i:128;9;49.

1. Стандартное равномерное распределение с заданным интервалом

Описание алгоритма:

Функция распределения $0 \leq x < 1: f(x) = 1$

Если стандартное равномерное случайное число U получено методом, установленным в предыдущем параграфе, то равномерное случайное число должно быть получено в соответствии со следующей формулой: $Y = b * U + a$.

Параметры запуска программы:

/d:st /p1:0 /p2:1

Исходный текст алгоритма:

```
def st(p1, p2, x):
    if x == 'Error':
        return x
    res = []
    left_border = 0
    right_border = len(x)
    for i in range(left_border, right_border):
        u = x[i] % 1024
        y = p2 * u + p1
        y = format(y, '.4f')
```

```
            res.append(y)
        return res
```

Результат работы программы:

2. Треугольное распределение

Описание алгоритма:

Функция распределения $a - b \leq x < a + b$: $f(x) = \frac{b-|a-x|}{b2}$

Если стандартные случайные числа $U1$ и $U2$ независимо получены методом генерации стандартного равномерного числа, то случайное число Y, подчиняющееся треугольному распределению, определяют по формуле: $Y = a + b * (U1 + U2 - 1)$.

Параметры запуска программы:

/d:tr /p1:0 /p2:1

Результат работы программы:

```
def tr(p1, p2, x):
    if x == 'Error':
        return x
    res = []
    left_border = 1
```

```
        right_border = len(x)
        for i in range(left_border, right_border):
            u1 = x[i - 1] % 1024
            u2 = x[i] % 1024
            y = p1 + p2 * (u1 + u2 - 1)
            y = format(y, '.4f')
            res.append(y)
        return res
```

Результат работы программы:

3. Общее экспоненциальное распределение

Описание алгоритма:

Функция распределения $x \geq a, f(x) = \frac{1}{b} * exp(-\frac{x-a}{b})$

Случайное число, соответствующее экспоненциальному распределению, получают по формуле: $Y = -b * ln(U) + a$.

Параметры запуска программы:

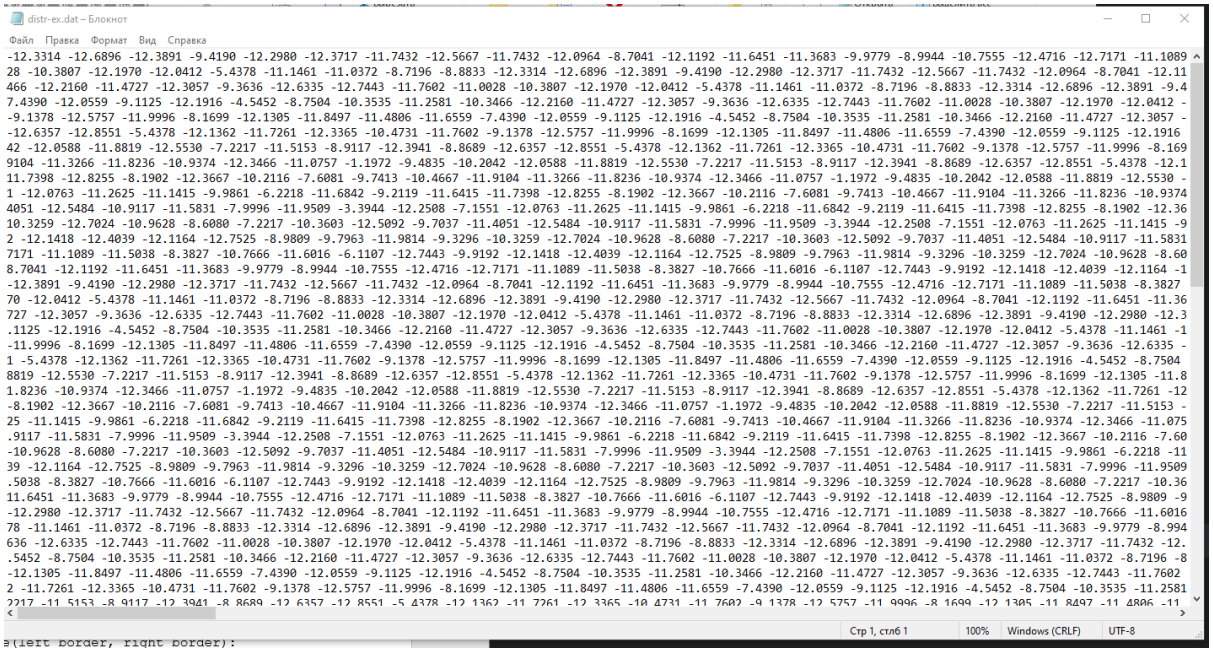
/d:ex /p1:1 /p2:2

Исходный текст алгоритма:

```
def ex(p1, p2, x):
```

```
    if x == 'Error':
        return x
    res = []
    left_border = 0
    right_border = len(x)
    for i in range(left_border, right_border):
        u = x[i] % 1024
        y = math.log1p(u) * (-p2) + p1
        y = format(y, '.4f')
        res.append(y)
    return res
```

Результат работы программы:



4. Нормальное распределение

Описание алгоритма:

Функция распределения $f(x) = \frac{1}{\sqrt{2\pi b}} exp\{-1/2b2(x-a)2\}$

Преобразование:

$$y1, y2 = nr(x1, x2, a, b): y1 = a + b * \sqrt{-2\ ln\ ln\left(1 - U(x1)\right)}\, cos(2\pi U(x2))y2$$

$$= a + b * \sqrt{-2\ ln\ ln\left(1 - U(x1)\right)}\, sin(2\pi U(x2))$$

Параметры запуска программы:

/d:nr /p1:1 /p2:2

Исходный текст алгоритма:

```
def nr(p1, p2, x):
    if x == 'Error':
        return x
    res = []
    left_border = 0
    right_border = len(x)
    step = 2
    for i in range(left_border, right_border, step):
        u1 = x[i] % 1024
        u2 = x[i + 1] % 1024
        z1 = p1 + p2 * math.sqrt(math.fabs((-2) *
math.log1p(math.fabs(1 - u1)))) * math.cos(2 * math.pi * u2)
        z2 = p1 + p2 * math.sqrt(math.fabs((-2) *
math.log1p(math.fabs(1 - u1)))) * math.sin(2 * math.pi * u2)
        z1 = format(z1, '.4f')
        z2 = format(z2, '.4f')
        res.append(z1)
        res.append(z2)
    return res
```

Результат работы программы:

5. Гамма распределение

Описание алгоритма:

Случайное число Y, подчиняющееся гамма распределению, определяют по формуле: $Y = a - bln\{(1 - U_i)(1 - U_{i+1})\}$.

Параметры запуска программы:

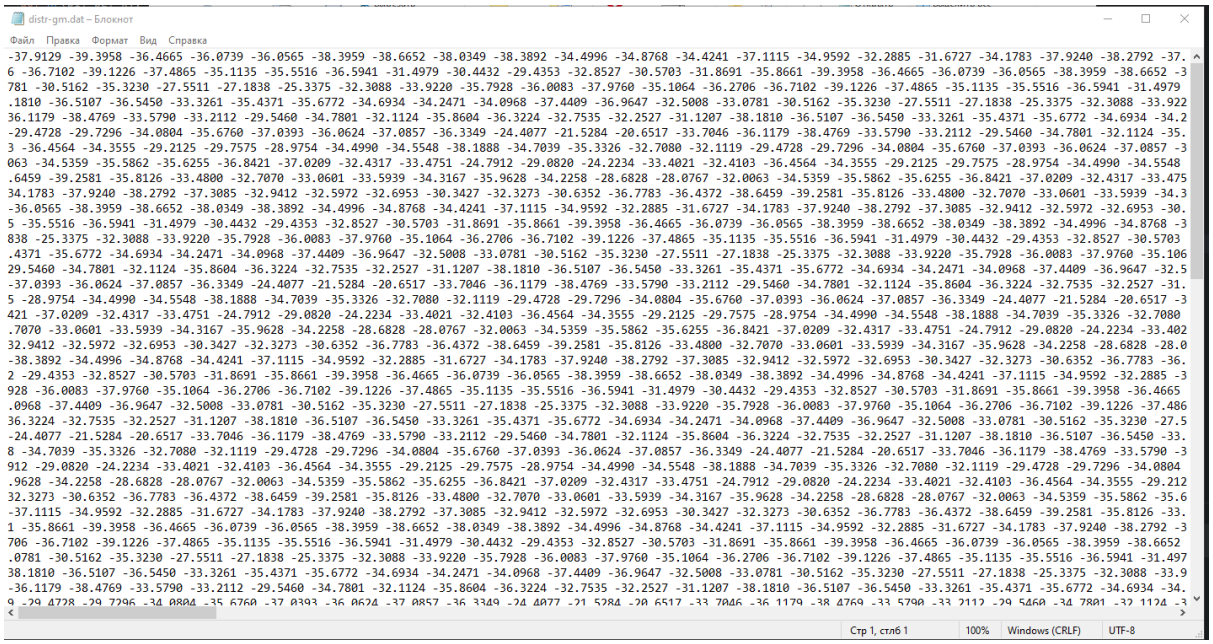
/d:gm /p1:2 /p2:3 /p3:3

Исходный текст алгоритма:

```
def gm(p1, p2, p3, x):
    if x == 'Error':
        return x
    res = []
    left_border = 1
    right_border = len(x)
    for i in range(left_border, right_border):
        u = []
        for j in range(p3):
            u.append(x[i - j] % 1024)
        u_mult = 1
        for j in range(len(u)):
            u_mult = u_mult * (1 - u[j])
        y = p1 - p2 * math.log1p(math.fabs(u_mult))
```

```
        y = format(y, '.4f')
        res.append(y)
    return res
```

Результат работы программы:



6. Логнормальное распределение

Описание алгоритма:

Функция распределения $f(x) = \frac{1}{\left(\sqrt{2\pi}\left\{\frac{x-a}{b}\right\}\right)expexp\left\{-\frac{1}{2}\left(\frac{x-a}{b}\right)2\right\}}$ при $x \geq a$

Преобразование:

$$y1, y2 = lognorm(x1, x2, a, b): z1, z2 = norm(x1, x2, 0{,}1, m)y1 = a + exp(b - z1)y2 = a + exp(b - z2)$$

Случайное число Y, подчиняющееся логнормальному распределению, определяют по формуле: $Y = a + exp(b - U)$.

Параметры запуска программы:

/d:ln /p1:1 /p2:2

Исходный текст алгоритма:

```
def ln(p1, p2, x):
    if x == 'Error':
```

```
        return x
    res = []
    left_border = 0
    right_border = len(x)
    for i in range(left_border, right_border):
        u = x[i] % 1024
        y = p1 + math.exp(p2 - u)
        y = format(y, '.4f')
        res.append(y)
    return res
```

Результат работы программы:

7. Логистическое распределение

Описание алгоритма:

Случайное число Y, подчиняющееся логистическому распределению, определяют по формуле:

$$Y = a + b * ln\left(\frac{U}{1-U}\right).$$

Параметры запуска программы:

/d:ls /p1:1 /p2:2

Исходный текст алгоритма:

```
def ls(p1, p2, x):
    if x == 'Error':
        return x
    res = []
    left_border = 0
    right_border = len(x)
    for i in range(left_border, right_border):
        u = x[i] % 1024
        y = p1 + p2 * math.log1p(math.fabs(u / (1 - u)))
        y = format(y, '.4f')
        res.append(y)
    return res
```

Результат работы программы:

8. Биномиальное распределение

Описание алгоритма:

Преобразование:

$y = binominal(x, a, b, m)$: $u = U(x)$ $s = 0$ $k = 0$ $loopstart$:

$s = s + Cbk\ ak\ (1 - a)b - k$ $if\ s > u$: $y = k$ Завершить

$if\ k < b - 1$: $k = k + 1$ перейти к loopstart

$y = b$

Коэффициент $C_b^k = \frac{b!}{k!(b-k)!}$

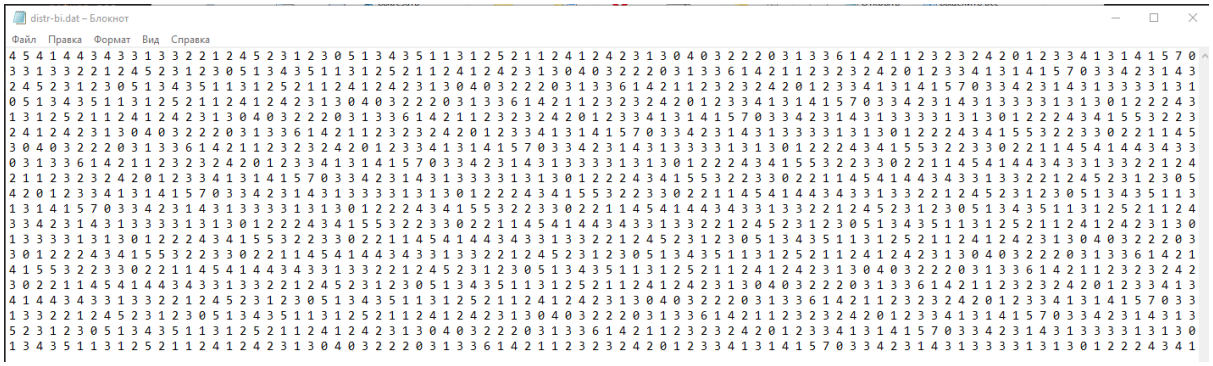
Параметры запуска программы:

/d:bi /p1:0.1 /p2:27

Исходный текст алгоритма:

```
def bi(p1, p2, x):
    if x == 'Error':
        return x
    res = []
    left_border = 0
    right_border = len(x)
    for i in range(left_border, right_border):
        u = x[i] / 1024
        y = 0
        s = 0
        k = 0
        while (True):
            s = s + (math.factorial(p2) / (math.factorial(k) * math.factorial(p2 - k))) * (p1 ** k) * ((1 - p1) ** (p2 - k))
            if s > u:
                y = k
                break
            if k < p2 - 1:
                k = k + 1
                continue
            y = p2
            break
        res.append(y)
    return res
```

Результат работы программы:


distr-bi.dat – Блокнот
Файл Правка Формат Вид Справка