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 и	ракультета компьютерных наук и информационных технологий														
Черватюка Сергея Александровича	a														
фамилия, имя, отчество															
Научный руководитель															
Ст. преподаватель		И.И. Слеповичев													
	полпись, дата														

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 $0 \le x < 1$: 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

```
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')
```

Alba Plasses (Popular Ring, Crippers

724, Agene 918, Depart Ring, Crippers

724, Depart Ring, Crippers

724, Depart Ring, Crippers

725, Depart Ring, Crippers

725, Depart Ring, Crippers

726, Depart Ring, Crippers

726, Depart Ring, Crippers

726, Depart Ring, Crippers

726, Depart Ring, Crippers

727, Depart Ring, Crippers

727, Depart Ring, Crippers

727, Depart Ring, Crippers

728, Depart Ring, Crippers

729, Depart Ring, Crippers

2. Треугольное распределение

Описание алгоритма:

Функция распределения
$$a-b \le x < a+b$$
: $f(x) = \frac{b-|a-x|}{b^2}$

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:

```
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
```

TYLL ROMO 1746, A0000 881, R000 1570, R000 1

3. Общее экспоненциальное распределение

Описание алгоритма:

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

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
```

Edit-codds Excesser Bug Copases

Cash Pages Capases

Li 2,3314 - 12,6566 - 12,3891 - 9,4199 - 12,2989 - 12,3717 - 11,7432 - 12,6667 - 11,7432 - 12,6667 - 11,7432 - 12,6667 - 11,7432 - 12,666 - 12,1575 - 11,9598 - 12,657 - 11,5775 - 11,9596 - 12,637 - 11,648 - 12,1575 - 11,9598 - 12,667 - 11,648 - 12,1575 - 11,9598 - 12,667 - 12,1575 - 11,9598 - 12,667 - 12,1575 - 11,9598 - 12,677 - 1

4. Нормальное распределение

Описание алгоритма:

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

$$y1, y2 = nr(x1, x2, a, b): y1$$

$$= a + b * \sqrt{-2 \ln \ln (1 - U(x1))} \cos(2\pi U(x2))y2$$

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

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

```
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
```

distrigmidal - Excessor

calso Figures Openur Big Cipases

calso Figures

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 $x \ge a$

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

```
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
```

| Compart | Postage | Compart | Comp

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

```
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
```

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:

<i>a</i>	distr-	-bi.da	rt – B	локно	Т																																												-	-			\times
Файл	п Г	Прав	ca (Форма	т Ви	д С	праві	ка																																													
4 5	4 :	1 4	4 3	4 3	3 1	3 3	2 2	2 1	2 4	5 :	2 3	1	2 3	0 5	5 1	3 4	. 3	5 1	1	3 1	2	5 2	1 :	1 2	4 1	2	4 2	3	1 3	0	4 0	3 2	2 2	2 0	3 :	1 3	3 6	1 .	4 2	1 1	2 3	2	3 2	4 2	0	1 2	3	3 4	1 3	1 4	11	5 7	0 ^
3 3	1	3 3	2 2	1 2	4 5	2 3	1 2	2 3	0 5	1	3 4	3	5 1	1 3	3 1	2 5	2	1 1	2	4 1	2 4	1 2	3 :	1 3	0 4	1 0	3 2	2	2 0	3	1 3	3 6	5 1	4 2	1	12	3 2	3	2 4	2 0	1 2	2 3	3 4	1 3	1	4 1	5	7 0	3 3	4 2	2 3 :	1 4	3
2 4	5 3	2 3	1 2	3 0	5 1	3 4	3 5	5 1	1 3	1 :	2 5	2	1 1	2 4	1 1	2 4	2	3 1	3	0 4	0	3 2	2 :	2 0	3 1	1 3	3 6	1	4 2	1	1 2	3 2	2 3	2 4	2 (9 1	2 3	3 .	4 1	3 1	4 1	5	7 0	3 3	4	2 3	1 4	4 3	1 3	3 3	3 3 :	1 3	1
0 5	1	3 4	3 5	1 1	3 1	2 5	2 1	1 1	2 4	1 :	2 4	2	3 1	3 (4	0 3	2	2 2	0	3 1	3	3 6	1 4	4 2	1 1	2	3 2	3	2 4	2	0 1	2 3	3	4 1	3 :	1 4	1 5	7	0 3	3 4	2 3	1 1	4 3	1 3	3	3 3	1	3 1	3 0	1 2	2 2	2 4	3
1 3	1 :	2 5	2 1	1 2	4 1	2 4	2 :	3 1	3 0	4 (0 3	2	2 2	0 3	3 1	3 3	6	1 4	2	1 1	2	3 2	3 :	2 4	2 6	1	2 3	3	4 1	3	1 4	1 5	7	0 3	3 4	4 2	3 1	4	3 1	3 3	3 3	1	3 1	3 0	1	2 2	2 4	4 3	4 1	5 5	5 3 :	2 2	3
2 4	1 :	2 4	2 3	1 3	0 4	0 3	2 2	2 2	0 3	1	3 3	6	1 4	2 :	1 1	2 3	2	3 2	4	2 0	1	2 3	3 4	4 1	3 1	4	1 5	7	0 3	3	4 2	3 1	4	3 1	3	3 3	3 1	3	1 3	0 1	2 2	2 2	4 3	4 1	. 5	5 3	2 :	2 3	3 0	2 2	21	1 4	5
3 0	4 (0 3	2 2	2 0	3 1	3 3	6 1	1 4	2 1	1 :	2 3	2	3 2	4 2	2 0	1 2	3	3 4	1	3 1	4	l 5	7 (3	3 4	1 2	3 1	. 4	3 1	. 3	3 3	3 1	L 3	1 3	0	1 2	2 2	4	3 4	1 5	5 3	2 :	2 3	3 6	2	2 1	1 4	4 5	4 1	4 4	13	4 3	3
0 3	1	3 3	6 1	4 2	1 1	2 3	2 :	3 2	4 2	0 :	1 2	3	3 4	1 3	3 1	4 1	. 5	7 0	3	3 4	2	3 1	4	3 1	3 3	3 3	3 1	3	1 3	0	1 2	2 2	2 4	3 4	1	5 5	3 2	2	3 3	0 2	2 1	114	4 5	4 1	4	4 3	4	3 3	1 3	3 2	2 2	1 2	4
2 1	1 :	2 3	2 3	2 4	2 0	1 2	3 3	3 4	1 3	1 4	4 1	5	7 0	3 3	3 4	2 3	1	4 3	1	3 3	3	3 1	3 :	1 3	0 1	2	2 2	4	3 4	1	5 5	3 2	2 2	3 3	0	2 2	1 1	4	5 4	1 4	4 3	4	3 3	1 3	3	2 2	1 :	2 4	5 2	3 1	1 2	3 0	5
4 2	0 :	1 2	3 3	4 1	3 1	4 1	5 7	7 0	3 3	4 :	2 3	1 4	4 3	1 3	3 3	3 3	1	3 1	3	0 1	2	2 2	4	3 4	1 5	5 5	3 2	2	3 3	0	2 2	1 1	. 4	5 4	1 4	4 4	3 4	3	3 1	3 3	2 2	1 :	2 4	5 2	3	1 2	3 (0 5	1 3	4 3	3 5	1 1	3
1 3	1 4	4 1	5 7	0 3	3 4	2 3	1 4	4 3	1 3	3	3 3	1	3 1	3 (1	2 2	2	4 3	4	1 5	5	3 2	2	3 3	0 2	2 2	1 1	. 4	5 4	1	4 4	3 4	1 3	3 1	3	3 2	2 1	2 .	4 5	2 3	1 2	3 (0 5	1 3	4	3 5	1 :	1 3	1 2	5 2	2 1	1 2	4
3 3	4	2 3	1 4	3 1	3 3	3 3	1 3	3 1	3 0	1 :	2 2	2 4	4 3	4 1	1 5	5 3	2	2 3	3	0 2	2	1 1	4 !	5 4	1 4	1 4	3 4	3	3 1	. 3	3 2	2 1	L 2	4 5	2	3 1	2 3	0	5 1	3 4	3 5	1:	1 3	1 2	5	2 1	1 2	2 4	1 2	4 2	2 3 :	1 3	0
1 3	3	3 3	1 3	1 3	0 1	2 2	2 4	4 3	4 1	5 5	5 3	2	2 3	3 6	2	2 1	1	4 5	4	1 4	4	3 4	3	3 1	3 3	3 2	2 1	2	4 5	2	3 1	2 :	3 0	5 1	3 4	4 3	5 1	1	3 1	2 5	2 1	1 :	2 4	1 2	4	2 3	1	3 0	4 0	3 2	2 2	2 0	3
3 0	1 :	2 2	2 4	3 4	1 5	5 3	2 2	2 3	3 0	2 :	2 1	1 4	4 5	4 1	. 4	4 3	4	3 3	1	3 3	2	2 1	2 4	4 5	2 :	3 1	2 3	0	5 1	. 3	4 3	5 1	1	3 1	2	5 2	1 1	2 .	4 1	2 4	2 3	1	3 0	4 6	3	2 2	2 (0 3	1 3	3 6	5 1	4 2	1
4 1	5 !	5 3	2 2	3 3	0 2	2 1	1 4	4 5	4 1	4 4	4 3	4	3 3	1 3	3 3	2 2	1	2 4	5	2 3	1	2 3	0 !	5 1	3 4	1 3	5 1	1	3 1	. 2	5 2	1 1	L 2	4 1	2 4	4 2	3 1	3	0 4	0 3	2 2	2 2	0 3	1 3	3	6 1	4 :	2 1	1 2	3 2	2 3	2 4	2
3 0	2 :	2 1	1 4	5 4	1 4	4 3	4	3 3	1 3	3 :	2 2	1	2 4	5 2	2 3	1 2	3	0 5	1	3 4	3 !	5 1	1	3 1	2 5	5 2	1 1	2	4 1	2	4 2	3 1	1 3	0 4	0	3 2	2 2	0	3 1	3 3	6 1	4	2 1	1 2	3	2 3	2 4	4 2	0 1	2 3	3 3	4 1	3
4 1	4	4 3	4 3	3 1	3 3	2 2	1 1	2 4	5 2	3 :	1 2	3 (a 5	1 3	3 4	3 5	1	1 3	1	2 5	2	1 1	2 4	4 1	2 4	1 2	3 1	3	0 4	0	3 2	2 2	0	3 1	3	3 6	1 4	2	1 1	2 3	2 3	2 /	4 2	0 1	2	3 3	4 :	1 3	1 4	1 5	5 7 (0 3	3
1 3	3 :	2 2	1 2	4 5	2 3	1 2	3 6	0 5	1 3	4	3 5	1	1 3	1 2	2 5	2 1	1	2 4	1	2 4	2	3 1	3 (9 4	0 3	3 2	2 2	0	3 1	3	3 6	1 4	1 2	1 1	2	3 2	3 2	4	2 0	1 2	3 3	4 :	1 3	1 4	1	5 7	0	3 3	4 2	3 1	1 4	3 1	3
5 2	3	1 2	3 0	5 1	3 4	3 5	1 1	1 3	1 2	5	2 1	1	2 4	1 2	4	2	1	3 0	4	03	2	2	0	3 1	3 :	3 6	1 4	2	1 1	2	3 2	3 2	4	2 0	1	2 3	3 4	1	3 1	4 1	5 7	10	3 3	4 2	3	1 4	3	1 3	3 3	3 1	1 3	1 3	0
1 3	4	3 5	1 1	3 1	2 5	2 1	1 2	2 4	1 2	4	2 3	1	3 0	4 6	3	2 2	2	03	1	33	6	14	2	1 1	2	3 2	3 2	4	2 0	1	23	3 4	1 1	3 1	4	1 5	7 0	3	3 4	2 3	1 4	1 3	1 3	3 3	3	1 3	1	3 0	1 2	2 7	2 4	34	1
1				_					_			-						_	_	_	-					_				-	_		_											-					_				