

Федеральное агентство связи
Ордена трудового Красного Знамени федеральное государственное
бюджетное
образовательное учреждение высшего образования
«Московский технический университет связи и информатики»

Отчет по лабораторной работе
«КОЛИЧЕСТВЕННАЯ ОЦЕНКА СТОЙКОСТИ ПАРОЛЬНОЙ ЗАЩИТЫ»
по дисциплине «Информационная безопасность»
Вариант 1

Выполнил: студент группы БВТ1904

Абакаров Гасан Гаджирабаданович

Проверила:

Магомедова Дженнет Исламутдиновна

Москва, 2020

Цель работы: получение основных теоретических сведений и практических навыков по оценке стойкости парольной защиты.

Ход работы:

1. Ознакомиться с теоретической частью данной работы.
2. Реализовать простейший генератор паролей, обладающий требуемой стойкостью к взлому.
3. Составить отчет по проделанной работе.
4. Защитить работу.

Постановка задачи:

В таблице 3 найти для вашего варианта значения характеристик P , V , T .

1. Вычислить по формуле $S^* = VT/P$ нижнюю границу S^* для заданных P , V , T .
2. Выбрать некоторый алфавит с мощностью A и получить минимальную длину пароля L , при котором выполняется условие $S^* \leq S = A^L$.
3. Реализовать программу-генератор паролей пользователей. Программа должна формировать случайную последовательность символов длины L , при этом должен использоваться алфавит из A символов.

Значения для варианта 1:

$P=0.0001$

$V=15$ паролей/мин

$T=2$ недели = 20160 минут

Листинг программы:

```
import math
import tkinter as tk
import tkinter.ttk as ttk
from tkinter import messagebox
import random
```

```

def ctrlEvent(e):
    if (e.state == 20 and e.keysym == 'c'): return
    return "break"

def callback(text, d=True):
    p=''
    for i in text.get().lower():
        if d:
            if i==',' : i='.'
            if i=='.' :
                p+='.'
                d=False
                continue
        if i in abc.numbers:
            p+=i
    text.set(p)

def check(s, a):
    i=0
    for t in s.container:
        i+=t.get()
    if i==0:
        s.container[a-1].set(1)
    s.update()

class message:
    def save(self):
        l=self.text1.get()
        if l:
            if float(l)>=1:
                messagebox.showinfo('', 'вероятность не может быть больше или
равна 1')
            return
        self.parent.update([self.text1.get(),
                             self.text2.get(),
                             self.text3.get(),
                             self.text4.get(),

```

```

        self.text5.get()])
    self.toplevel.destroy()
def __init__(self, top=None):
    self.toplevel=top
    '''This class configures and populates the toplevel window.
       top is the toplevel containing window.'''

    self.text1 = tk.StringVar()
    self.text1.trace("w", lambda *a: callback(self.text1))

    self.text2 = tk.StringVar()
    self.text2.trace("w", lambda *a: callback(self.text2, False))

    self.text3 = tk.StringVar()
    self.text3.trace("w", lambda *a: callback(self.text3, False))

    self.text4 = tk.StringVar()
    self.text4.trace("w", lambda *a: callback(self.text4, False))

    self.text5 = tk.StringVar()
    self.text5.trace("w", lambda *a: callback(self.text5, False))

    _bgcolor = '#d9d9d9' # X11 color: 'gray85'
    _fgcolor = '#000000' # X11 color: 'black'
    _compcolor = '#d9d9d9' # X11 color: 'gray85'
    _ana1color = '#d9d9d9' # X11 color: 'gray85'
    _ana2color = '#ececec' # Closest X11 color: 'gray92'
    self.style = ttk.Style()
    self.style.configure('.', background=_bgcolor)
    self.style.configure('.', foreground=_fgcolor)
    self.style.configure('.', font="TkDefaultFont")
    self.style.map('.', background=
        [('selected', _compcolor), ('active', _ana2color)])

    top.geometry("565x215+603+272")
    top.minsize(1, 1)
    top.maxsize(1905, 1050)

```

```

top.resizable(1, 1)
top.title("")
top.configure(highlightcolor="black")

self.Frame1 = tk.Frame(top)
self.Frame1.place(relx=0.018, rely=0.047, relheight=0.907
                  , relwidth=0.965)
self.Frame1.configure(relief='groove')
self.Frame1.configure(borderwidth="2")
self.Frame1.configure(relief="groove")

self.TEntry1 = ttk.Entry(self.Frame1, textvariable=self.text1)
self.TEntry1.place(relx=0.495, rely=0.051, relheight=0.108
                  , relwidth=0.484)

self.TEntry1.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TEntry1.configure(takefocus="")
self.TEntry1.configure(cursor="xterm")

self.TLabel1 = ttk.Label(self.Frame1)
self.TLabel1.place(relx=0.018, rely=0.051, height=26, width=106)
self.TLabel1.configure(background="#d9d9d9")
self.TLabel1.configure(foreground="#000000")
self.TLabel1.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel1.configure(relief="flat")
self.TLabel1.configure(anchor='w')
self.TLabel1.configure(justify='left')
self.TLabel1.configure(text='''Вероятность''')

self.TLabel1_3 = ttk.Label(self.Frame1)
self.TLabel1_3.place(relx=0.018, rely=0.256, height=26, width=246)
self.TLabel1_3.configure(background="#d9d9d9")
self.TLabel1_3.configure(foreground="#000000")
self.TLabel1_3.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel1_3.configure(relief="flat")
self.TLabel1_3.configure(anchor='w')

```

```

self.TLabel1_3.configure(justify='left')
self.TLabel1_3.configure(text='''Скорость перебора (паролей/мин)''')

self.TEntry1_4 = ttk.Entry(self.Frame1, textvariable=self.text2)
self.TEntry1_4.place(relx=0.495, rely=0.256, relheight=0.108
                    , relwidth=0.484)

self.TEntry1_4.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TEntry1_4.configure(takefocus="")
self.TEntry1_4.configure(cursor="xterm")

self.TLabel1_4 = ttk.Label(self.Frame1)
self.TLabel1_4.place(relx=0.018, rely=0.615, height=26, width=246)
self.TLabel1_4.configure(background="#d9d9d9")
self.TLabel1_4.configure(foreground="#000000")
self.TLabel1_4.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel1_4.configure(relief="flat")
self.TLabel1_4.configure(anchor='w')
self.TLabel1_4.configure(justify='left')
self.TLabel1_4.configure(text='''Срок действия пароля''')

self.TEntry1_5 = ttk.Entry(self.Frame1, textvariable=self.text3)
self.TEntry1_5.place(relx=0.404, rely=0.615, relheight=0.108
                    , relwidth=0.172)

self.TEntry1_5.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TEntry1_5.configure(takefocus="")
self.TEntry1_5.configure(cursor="xterm")

self.TLabel2 = ttk.Label(self.Frame1)
self.TLabel2.place(relx=0.459, rely=0.513, height=17, width=35)
self.TLabel2.configure(background="#d9d9d9")
self.TLabel2.configure(foreground="#000000")
self.TLabel2.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel2.configure(relief="flat")
self.TLabel2.configure(anchor='w')
self.TLabel2.configure(justify='left')

```

```

self.TLabel2.configure(text='''дней''')

self.TLabel2_7 = ttk.Label(self.Frame1)
self.TLabel2_7.place(relx=0.661, rely=0.513, height=17, width=44)
self.TLabel2_7.configure(background="#d9d9d9")
self.TLabel2_7.configure(foreground="#000000")
self.TLabel2_7.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel2_7.configure(relief="flat")
self.TLabel2_7.configure(anchor='w')
self.TLabel2_7.configure(justify='left')
self.TLabel2_7.configure(text='''часов''')

self.TLabel2_8 = ttk.Label(self.Frame1)
self.TLabel2_8.place(relx=0.862, rely=0.513, height=17, width=54)
self.TLabel2_8.configure(background="#d9d9d9")
self.TLabel2_8.configure(foreground="#000000")
self.TLabel2_8.configure(font="-family {DejaVu Sans} -size 10 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TLabel2_8.configure(relief="flat")
self.TLabel2_8.configure(anchor='w')
self.TLabel2_8.configure(justify='left')
self.TLabel2_8.configure(text='''минут''')

self.TEntry1_2 = ttk.Entry(self.Frame1, textvariable=self.text4)
self.TEntry1_2.place(relx=0.624, rely=0.615, relheight=0.108
, relwidth=0.154)
self.TEntry1_2.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TEntry1_2.configure(takefocus="")
self.TEntry1_2.configure(cursor="xterm")

self.TEntry1_3 = ttk.Entry(self.Frame1, textvariable=self.text5)
self.TEntry1_3.place(relx=0.826, rely=0.615, relheight=0.108
, relwidth=0.154)
self.TEntry1_3.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.TEntry1_3.configure(takefocus="")
self.TEntry1_3.configure(cursor="xterm")

```

```

self.Label1 = tk.Label(self.Frame1)
self.Label1.place(relx=0.569, rely=0.615, height=22, width=27)
self.Label1.configure(activebackground="#f9f9f9")
self.Label1.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label1.configure(text=''+''')

self.Label1_4 = tk.Label(self.Frame1)
self.Label1_4.place(relx=0.771, rely=0.615, height=22, width=27)
self.Label1_4.configure(activebackground="#f9f9f9")
self.Label1_4.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label1_4.configure(text=''+''')

self.Button1 = tk.Button(self.Frame1)
self.Button1.place(relx=0.771, rely=0.769, height=28, width=115)
self.Button1.configure(activebackground="#f9f9f9")
self.Button1.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Button1.configure(text='''Сохранить''')
self.Button1.configure(command=self.save)

```

```

class abc:
    abc=[
        'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
        'abcdefghijklmnopqrstuvwxyz',
        "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ",
        "абвгдеёжзийклмнопрстуфхцчшщъыьэюя",
        '0123456789',
        '! " # $ % & \ ' ( ) * '
    ]
    len=[26, 26, 33, 33, 10, 10]
    def generate_password(self, len, abc_order):
        t=''
        while len>0:
            t+=random.choice(self.abc[random.choice(abc_order)])

```



```

        len-=1
    return t
abc=abc()

class Form:
    def update(self, c=None):
        if c!=None:
            if not c[0] in ('', '.'):
                self.p=float(c[0])
                self.label_p.configure(text=c[0])
            if c[1]:
                self.v=int(c[1])
                self.label_v.configure(text=c[1]+' паролей/мин')
            c1=int('0'+c[2])
            c2=int('0'+c[3])
            c3=int('0'+c[4])
            t=0
            if c1:
                t+=int(c[2])*1440
            if c2:
                t+=int(c[3])*60
            if c3:
                t+=int(c[4])
            if t:
                self.t=t
                self.label_t.configure(text=str(t)+' минут')
        i=0
        l=self.container
        self.a=0
        self.abc=[]
        while i<6:
            z=l[i].get()
            if z: self.abc.append(i)
            self.a+=z*abc.len[i]
            i+=1
        self.s=int(self.v*self.t/self.p)
        self.l=int(math.log(self.s, self.a))+1

```

```

self.label_a.configure(text=str(self.a))
self.label_s.configure(text=str(self.s))
self.label_l.configure(text=str(self.l))

def __init__(self, top=None):
    '''This class configures and populates the toplevel window.
        top is the toplevel containing window.'''
    _bgcolor = '#d9d9d9' # X11 color: 'gray85'
    _fgcolor = '#000000' # X11 color: 'black'
    _compcolor = '#d9d9d9' # X11 color: 'gray85'
    _ana1color = '#d9d9d9' # X11 color: 'gray85'
    _ana2color = '#ecec' # Closest X11 color: 'gray92'
    self.style = ttk.Style()
    self.style.configure('.', background=_bgcolor)
    self.style.configure('.', foreground=_fgcolor)
    self.style.configure('.', font="TkDefaultFont")
    self.style.map('.', background=
        [('selected', _compcolor), ('active', _ana2color)])
    self.toplevel=top
    self.che1=tk.IntVar()
    self.che1.set(1)
    self.che2=tk.IntVar()
    self.che3=tk.IntVar()
    self.che4=tk.IntVar()
    self.che5=tk.IntVar()
    self.che6=tk.IntVar()

    self.p=0.0001
    self.v=15
    self.t=20160

    self.container=[self.che1, self.che2, self.che3, self.che4,
self.che5, self.che6]

    top.geometry("911x379+510+251")
    top.minsize(1, 1)

```

```

top.maxsize(1905, 1050)
top.resizable(1, 1)
top.title("")
top.configure(highlightcolor="black")

self.Frame1 = tk.Frame(top)
self.Frame1.place(relx=0.022, rely=0.053, relheight=0.673
    , relwidth=0.576)
self.Frame1.configure(relief='groove')
self.Frame1.configure(borderwidth="2")
self.Frame1.configure(relief="groove")

self.Label1 = tk.Label(self.Frame1)
self.Label1.place(relx=0.019, rely=0.039, height=22, width=134)
self.Label1.configure(activebackground="#f9f9f9")
self.Label1.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label1.configure(text='''P (вероятность)''')

self.Label2 = tk.Label(self.Frame1)
self.Label2.place(relx=0.019, rely=0.157, height=22, width=184)
self.Label2.configure(activebackground="#f9f9f9")
self.Label2.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label2.configure(text='''V (скорость перебора)''')

self.Label3 = tk.Label(self.Frame1)
self.Label3.place(relx=0.019, rely=0.275, height=22, width=204)
self.Label3.configure(activebackground="#f9f9f9")
self.Label3.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label3.configure(text='''T (срок действия пароля)''')

self.label_p = tk.Label(self.Frame1, anchor='w')
self.label_p.place(relx=0.571, rely=0.039, height=22, width=200)
self.label_p.configure(activebackground="#f9f9f9")
self.label_p.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.label_p.configure(text='''0.0001''')

```

```

self.label_v = tk.Label(self.Frame1, anchor='w')
self.label_v.place(relx=0.571, rely=0.157, height=22, width=200)
self.label_v.configure(activebackground="#f9f9f9")
self.label_v.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.label_v.configure(text='''15 паролей/мин''')

self.label_t = tk.Label(self.Frame1, anchor='w')
self.label_t.place(relx=0.571, rely=0.275, height=22, width=200)
self.label_t.configure(activebackground="#f9f9f9")
self.label_t.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.label_t.configure(text='''20160 минут''')

self.Label4 = tk.Label(self.Frame1)
self.Label4.place(relx=0.019, rely=0.471, height=22, width=244)
self.Label4.configure(activebackground="#f9f9f9")
self.Label4.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label4.configure(text='''S* (нижняя граница паролей)''')

self.Label_a = tk.Label(self.Frame1, anchor='w')
self.Label_a.place(relx=0.019, rely=0.588, height=22, width=200)
self.Label_a.configure(activebackground="#f9f9f9")
self.Label_a.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label_a.configure(text='''A (мощность алфавита)''')

self.Label_5 = tk.Label(self.Frame1)
self.Label_5.place(relx=0.019, rely=0.706, height=22, width=144)
self.Label_5.configure(activebackground="#f9f9f9")
self.Label_5.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Label_5.configure(text='''L (длина пароля)''')

self.label_s = tk.Label(self.Frame1, anchor='w')
self.label_s.place(relx=0.571, rely=0.471, height=22, width=200)
self.label_s.configure(activebackground="#f9f9f9")

```

```

        self.label_s.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.label_s.configure(text='')

        self.Button2 = tk.Button(self.Frame1)
        self.Button2.place(relx=0.648, rely=0.824, height=28, width=165)
        self.Button2.configure(activebackground="#f9f9f9")
        self.Button2.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.Button2.configure(text='''Изменить данные''')
        self.Button2.configure(command=lambda:
message(tk.Toplevel(root)).__setattr__('parent', self))

        self.label_a = tk.Label(self.Frame1, anchor='w')
        self.label_a.place(relx=0.571, rely=0.588, height=22, width=200)
        self.label_a.configure(activebackground="#f9f9f9")
        self.label_a.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")

        self.label_l = tk.Label(self.Frame1, anchor='w')
        self.label_l.place(relx=0.571, rely=0.706, height=22, width=200)
        self.label_l.configure(activebackground="#f9f9f9")
        self.label_l.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")

        self.Frame2 = tk.Frame(top)
        self.Frame2.place(relx=0.626, rely=0.053, relheight=0.673
, relwidth=0.346)
        self.Frame2.configure(relief='groove')
        self.Frame2.configure(borderwidth="2")
        self.Frame2.configure(relief="groove")

        self.check1 = tk.Checkbutton(self.Frame2)
        self.check1.place(relx=0.032, rely=0.039, relheight=0.094
, relwidth=0.654)
        self.check1.configure(activebackground="#f9f9f9")
        self.check1.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.check1.configure(justify='left')

```

```

self.check1.configure(text='''Латинские прописные''')
self.check1.configure(variable=self.ch1)

self.check2 = tk.Checkbutton(self.Frame2)
self.check2.place(relx=0.032, rely=0.157, relheight=0.094
                  , relwidth=0.622)
self.check2.configure(activebackground="#f9f9f9")
self.check2.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.check2.configure(justify='left')
self.check2.configure(text='''Латинские строчные''')
self.check2.configure(variable=self.ch2)

self.check3 = tk.Checkbutton(self.Frame2)
self.check3.place(relx=0.032, rely=0.275, relheight=0.094
                  , relwidth=0.59)
self.check3.configure(activebackground="#f9f9f9")
self.check3.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.check3.configure(justify='left')
self.check3.configure(text='''Русские прописные''')
self.check3.configure(variable=self.ch3)

self.check4 = tk.Checkbutton(self.Frame2)
self.check4.place(relx=0.032, rely=0.392, relheight=0.094
                  , relwidth=0.559)
self.check4.configure(activebackground="#f9f9f9")
self.check4.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
self.check4.configure(justify='left')
self.check4.configure(text='''Русские строчные''')
self.check4.configure(variable=self.ch4)

self.check5 = tk.Checkbutton(self.Frame2)
self.check5.place(relx=0.032, rely=0.51, relheight=0.094,
relwidth=0.273)

self.check5.configure(activebackground="#f9f9f9")

```

```

        self.check5.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.check5.configure(justify='left')
        self.check5.configure(text='''Цифры''')
        self.check5.configure(variable=self.che5)

        self.check6 = tk.Checkbutton(self.Frame2)
        self.check6.place(relx=0.032, rely=0.627, relheight=0.094
            , relwidth=0.94)
        self.check6.configure(activebackground="#f9f9f9")
        self.check6.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.check6.configure(justify='left')
        self.check6.configure(text='''Символы{!, ", #, $, %, &, ', (, ),
*}''')
        self.check6.configure(variable=self.che6)

        self.Frame3 = tk.Frame(top)
        self.Frame3.place(relx=0.022, rely=0.765, relheight=0.198,
relwidth=0.95)

        self.Frame3.configure(relief='groove')
        self.Frame3.configure(borderwidth="2")

        self.Label3 = tk.Label(self.Frame3)
        self.Label3.place(relx=0.012, rely=0.133, height=22, width=74)
        self.Label3.configure(activebackground="#f9f9f9")
        self.Label3.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.Label3.configure(text='''Пароль:''')

        self.password = ttk.Entry(self.Frame3)
        self.password.place(relx=0.104, rely=0.147, relheight=0.267
            , relwidth=0.883)
        self.password.configure(font="-family {DejaVu Sans} -size 12 -weight
normal -slant roman -underline 0 -overstrike 0")
        self.password.configure(takefocus="")
        self.password.configure(cursor="xterm")
        self.password.bind("<Key>", lambda e: ctrlEvent(e))

```

```

self.Button1 = tk.Button(self.Frame3)
self.Button1.place(relx=0.012, rely=0.533, height=28, width=205)
self.Button1.configure(activebackground="#f9f9f9")
self.Button1.configure(font="-family {DejaVu Sans} -size 11 -weight
normal -slant roman -underline 0 -overstrike 0")
self.Button1.configure(text='''Сгенерировать пароль''')
self.Button1.configure(command=lambda : (
                                self.password.delete('0',
'end'),
                                self.password.insert('0',
abc.generate_password(self.l, self.abc))
                                ))

```

```

self.check1.configure(command=lambda :check(self, 1))
self.check2.configure(command=lambda :check(self, 2))
self.check3.configure(command=lambda :check(self, 3))
self.check4.configure(command=lambda :check(self, 4))
self.check5.configure(command=lambda :check(self, 5))
self.check6.configure(command=lambda :check(self, 6))
self.update()

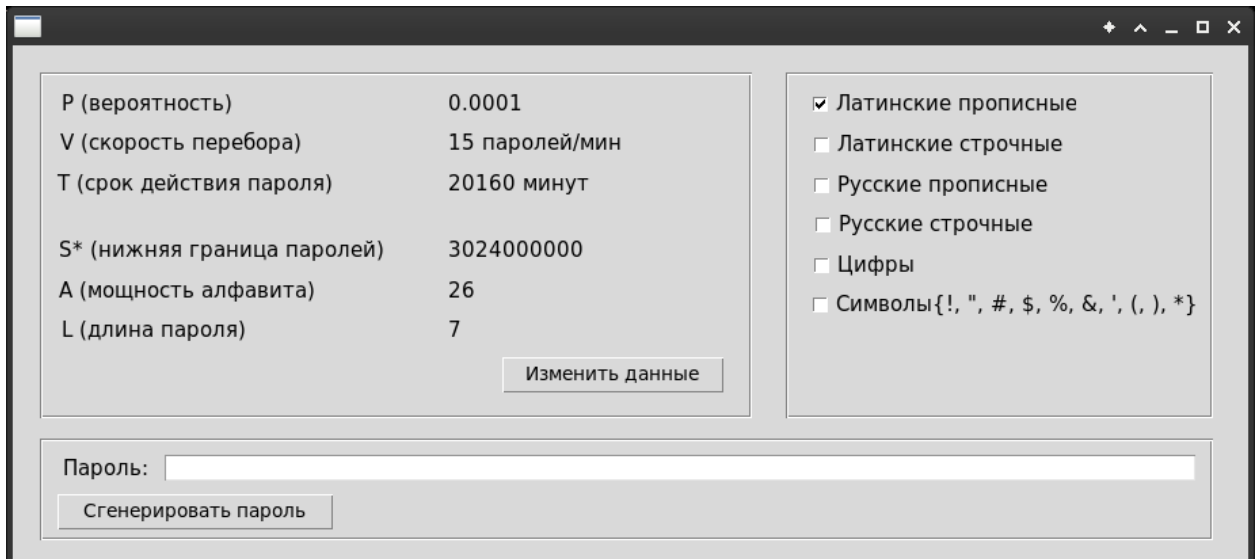
```

```

if __name__ == '__main__':
    root=tk.Tk()
    w=Form(root)
    w.toplevel.mainloop()

```


Результат выполнения программы:



The screenshot shows a window titled "Результат выполнения программы:" with a standard Windows-style title bar. The main area is divided into two panels. The left panel contains a list of parameters and their values: P (вероятность) 0.0001, V (скорость перебора) 15 паролей/мин, T (срок действия пароля) 20160 минут, S* (нижняя граница паролей) 3024000000, A (мощность алфавита) 26, and L (длина пароля) 7. Below this list is a button labeled "Изменить данные". The right panel contains a list of checkboxes for character sets: "Латинские прописные" (checked), "Латинские строчные" (unchecked), "Русские прописные" (unchecked), "Русские строчные" (unchecked), "Цифры" (unchecked), and "Символы{!, ", #, \$, %, &, ', (,), *}" (unchecked). At the bottom of the window is a text input field labeled "Пароль:" and a button labeled "Сгенерировать пароль".

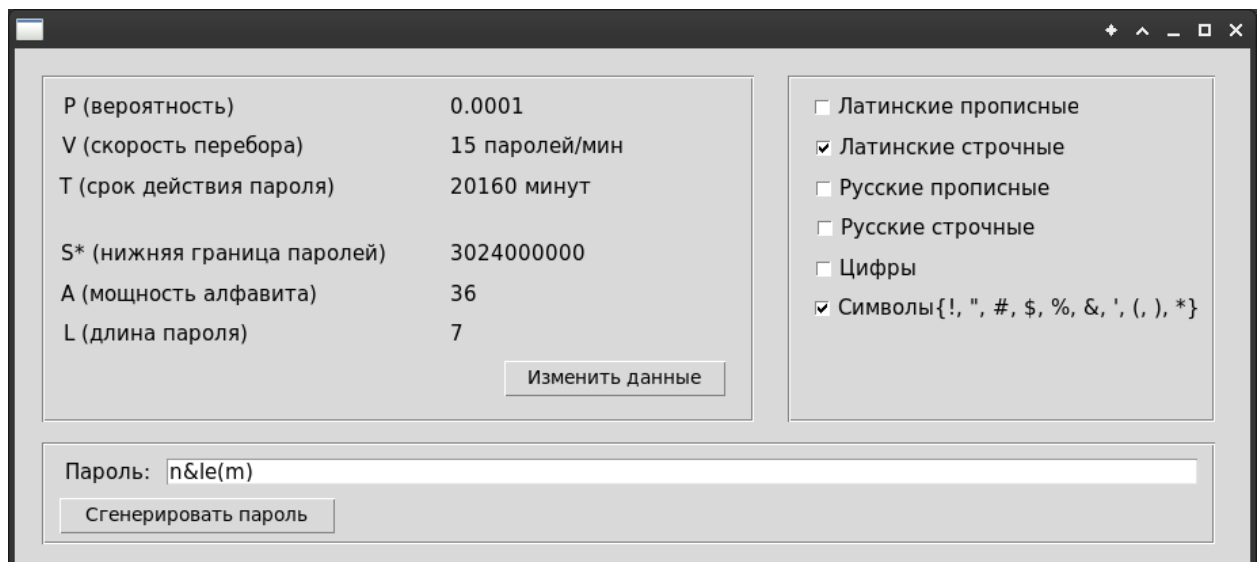
P (вероятность)	0.0001
V (скорость перебора)	15 паролей/мин
T (срок действия пароля)	20160 минут
S* (нижняя граница паролей)	3024000000
A (мощность алфавита)	26
L (длина пароля)	7

Изменить данные

Пароль:

Сгенерировать пароль

Рис 1. Основное окно приложения



This screenshot shows the same application window as in Figure 1, but with the "Сгенерировать пароль" button clicked. The "Пароль:" text input field now contains the generated password "n&le(m)". The configuration settings and checkboxes remain the same as in the previous screenshot.

P (вероятность)	0.0001
V (скорость перебора)	15 паролей/мин
T (срок действия пароля)	20160 минут
S* (нижняя граница паролей)	3024000000
A (мощность алфавита)	36
L (длина пароля)	7

Изменить данные

Пароль: n&le(m)

Сгенерировать пароль

Рис 2. Пример генерации пароля с латинскими строчными буквами и символами

Рис 3. Пример окна для изменения степени стойкости парольной защиты

Контрольные вопросы

1. Дать определение стойкости пароля к взлому. Написать формулу.
2. Дать определение мощности алфавита паролей.
3. Перечислить основные задачи, которые могут решаться с использованием определения стойкости пароля.
4. Перечислить основные требования к выбору пароля.

Ответы

1. Стойкость пароля к взлому — это вероятность подбора пароля злоумышленником в течении срока его действия, вычисляется по формуле:

$$P = \frac{VT}{A^L}, \text{ где } A — \text{мощность алфавита паролей, } L — \text{длина пароля, } S = AL$$

— число всевозможных паролей длины L , которые можно составить из символов алфавита A , V — скорость перебора паролей злоумышленником, T — максимальный срок действия пароля.

2. Мощность алфавита паролей - количество символов, которые могут быть использованы при составлении пароля

3. Основные задачи, которые могут решаться с использованием определения стойкости пароля — это проектирование и реализация программного обеспечения систем аутентификации.

4. Основные требования к выбору пароля:

Длина пароля (количество символов в пароле) не меньше минимальной длины.

Пароль не должен содержать трех и более одинаковых символов подряд.

Пароль не должен содержать общеупотребительные слова, имена, названия предметов.

Пароль не должен содержать последовательности, пароль должен иметь уникальную (случайную) комбинацию символов