UNIVERSIDADE FEDERAL DO PARANÁ

GRADUAÇÃO EM MATEMÁTICA

Ana Cecília Gheno da Silva
Gabriele Henrique Pedro
Thais Spannenberg Machado dos Passos

Controle de vendas do Centro Acadêmico

CURITIBA

2018

Ana Cecília Gheno da Silva - GRR20170390
Gabriele Henrique Pedro - GRR20170398
Thais Spannenberg Machado dos Passos - GRR20185667

Controle de vendas do Centro Acadêmico

Relatório apresentado à disciplina Fundamentos de Programação de Computadores do Curso de Graduação em Matemática da Universidade Federal do Paraná.

Orientador: Prof. Jackson Antônio do Prado Lima

Curitiba, 26 de Novembro de 2018.

SUMÁRIO

1 INTRODUÇÃO

Este relatório apresenta informações relativas ao trabalho final da matéria de Fundamentos de Programação de Computadores (CI182-MAT1), nosso objeto de estudo é a linguagem python que foi utilizada na construção e desenvolvimento de todo o trabalho.

A escolha de fazer esse programa surgiu da vontade de criar algo que tivesse utilidade após a finalização do projeto e, como estamos envolvidas na chapa vigente do Centro Acadêmico de Matemática, acreditamos que o nosso programa poder ser uma ferramenta importante para tesouraria.

2 OBJETIVOS

2.1 OBJETIVOGERAL

Nosso projeto tem como objetivo o gerenciamento da parte comercial e financeira do CAM (Centro Acadêmico de Matemática). Nosso programa apresentará funcionalidades divididas em partes interna e externa.

2.2 OBJETIVO ESPECÍFICO

- Habilitar a parte externa, para que quando acessada permita ao usuário a visualização da tabela de preços dos produtos comercializados pelo centro acadêmico.

-Permitir por meio de login e senha, o acesso da tabela de gerenciamento da contabilidade do centro acadêmico.

1 DESENVOLVIMENTO

1.1 OTrabalho

```
1 # -*- coding: utf-8 -*-
2 import pandas as pd
3 import tkinter as tk
4 from tkinter import *
5
6 #Comandos para configuração da telas
7 root= Tk() #Comandos principa
8 text= tk.Text()
```

A primeira linha do programa é usada para definir/declarar em qual código python o programa abaixo está escrito. Em outras palavras, é essa linha que irá dizer ao interpretador qual codificação dentro da linguagem python foi usada. As próximas três linhas são usadas para importar as bibliotecas que serão utilizadas posteriormente. Nas linhas 7 e 8 definimos as variáveis para que não precisemos escrever todo o caminho percorrido até as funções todas as vezes.[1]

```
12 def tela_inicial():
13     root.resizable(0, 0) #Edita o tamanho da tel
14
15    #Leitura e "print" da tabela externa
16     tabela= pd.read_table('Preco.csv', sep=",")
17     text.insert(tk.END, str(tabela))
18     text.config(state='disabled')
19     text.pack()
```

A função tela_inicial é a que edita a primeira tela do programa. Nessa função temos a chamada do comando root com a função resizable. Essa função é a responsável pela definição de mudança de tamanho da tela criada, nesse caso,os zeros são colocados, pois em python 3 eles tem o valor de Falso. A variável tabela é responsável pela leitura do arquivo “Preco.csv” e recebe o comando ‘sep= ”,” ’, pois os argumentos dentro do arquivo estão separados por vírgulas. O comando text é chamado com as funções insert, config e pack. Essas funções são, respectivamente, responsáveis pela adição

dos dados adquiridos na variável tabela, pela configuração de edição desabilitada e pela impressão dos dados na tela.[1]

```
22 def segunda_tela():
23     master= Tk() #Abre a tela
24     Label(master, text="Salvar alterações feitas").grid(row=1,)
25     master.title("Encerrar edição") #Dá nome a tela
26     Button(master, text='Encerrar', fg="blue", command= quit).grid(row=2,
27     | column=1, sticky=W, pady=4)
28 
29     #Edita a tabela
30     tabela= pd.read_table('Preco.csv', sep=",")
31     text.insert(tk.END, str(tabela))
32     text.config(state='normal')
33     text.pack()
```

A função segunda_tela configura a tela principal do programa pela segunda vez e abre uma tela suporte. A primeira linha da função define o comando principal que será usado para a abertura dessa tela. Da linha 24 até a linha 27, temos as configurações da tela pequena que acompanha a segunda tela. A parte Label (linha 24) define o texto que será impresso na tela e sua posição. A linha 25 define o título da tela suporte e as linhas 26 e 27 definem o botão que aparece nessa telinha, o local em que ele estará posicionado, seu tamanho, o texto escrito nele na cor azul e o comando de saída (anteriormente, além de fechar o programa seria esse botão o responsável por salvar as alterações feitas).
A segunda parte (das linhas 30 a 33) edita a parte principal dessa função. Os comandos são exatamente os mesmos da função tela_principal a única mudança é que agora a tabela na tela pode ser editada. Portanto, não é aberta uma nova tela, a primeira tela é editada.[1]

```
36 def login():
37     master = Tk() #Abre a tela
38     master.title("Acesso") #Dá nome a tela
39 
40     #Escrita na tela
41     Label(master, text="Login: ").grid(row=0)
42     Label(master, text="Senha: ").grid(row=1)
43 
44     #Entradas e telinhas brancas
45     login = Entry(master)
46     senha = Entry(master)
47     login.grid(row=0, column=1)
48     senha.grid(row=1, column=1)
49 
50     Button(master, text='Entrar', fg="blue",
51         command= lambda: confere_salva(senha.get(), login.get())).grid(
52         row=3, column=1, sticky=W, pady=4)
```

Nessa função teremos a criação da tela de login do programa. Definimos o comando principal e o título nas duas primeiras linhas. As linhas 41 e 42 são as responsáveis pelos textos “Login:” e “Senha:” que aparecem na tela recém criada. Entre as linhas 45 e 48 temos a criação das variáveis para salvar o conteúdo que será informado nas caixas brancas (espaço utilizado pelo usuário para entrar com informações) que também são criadas nesse intervalo de linhas. A mudança do último botão para este é o comando que agora é lido como lambda e uma função. Isso acontece porque ao apertar do botão queremos que a função confere_salva seja executada. Dentro da função temos o comando “.get()” que é o responsável por ler as informações dentro das caixas brancas. No decorrer dafunção podemos ver o comando “.grid()” que é o responsável por editar a localização das informações na tela criada.[1]

```
56 def encriptar(senha):
57   cipher = ''
58   for char in senha:
59     if char == ' ':
60       cipher = cipher + char
61     elif  char.isupper():
62       cipher = cipher + chr((ord(char) + 14 - 65) % 26 + 65)
63     else:
64       cipher = cipher + chr((ord(char) + 14 - 97) % 26 + 97)
65   return cipher
```

Essa é a função responsável por encriptar a senha recém adicionada pelo usuário para que possa ocorrer a conferência dessa com a senha já pré-estabelecida pelo programa. Essa é uma função já criada que pode ser encontrada no Google ao se pesquisar “CaesarCipherpython3”. Porém sua

função é basicamente embaralhar as letras tornando a palavra um código de novas letras. Para o nosso programa escolhemos o número 14, então a letra A vira O, B-P, C-Q e assim por diante.[1]

```
69 def confere_salva(senha, login):
70     master= Tk() #Abre a tela
71     entradas.append(login) #Colocar os logins na lista externa
72 
73     #Abre o arquivo com o código para comparação com a senha
74     with open('Codigo', 'r') as c:
75         codigo= c.read()
76         #Verifica a senha codificada
77         if str(codigo) == encriptar(senha):
78             segunda_tela()
79         else:
80             master.title("Ops!")
81             master.resizable(0,0) #Edita o tamanho da tela
82             Label(master, text="Tente novamente").grid(row=0, column= 1)
```

```
83 
84     #Salvando os logins
85     with open ('Usuário.txt', 'a', encoding='utf-8') as f:
86         f.write("Login: ")
87         f.write(login)
88         f.write(" - ")
89         f.write("Senha: ")
90         f.write(senha)
91         f.write("//")
```

Essa função responsável por conferir se a senha informada está correta e por salvar o login e a senha inseridos para futuras consultas. Além disso, essa função cria, quando a senha inserida está incorreta, uma tela com uma mensagem de erro. Começamos abrindo a tela e definindo o comando master. A linha 71 acabou não sendo removida, porém ela não foi utilizada e só salva os logins em uma lista no programa principal. Entre as linhas 74 e 82 temos a leitura do arquivo “Codigo” que possui a senha pré-estabelecida já encriptada, a criação de uma variável pra salvar essa informação e a conferência das duas encriptações para ver se a senha informada está certa. Caso a senha esteja certa, a função segunda_tela é chamada, caso contrário é criada a tela com mensagem de erro. O fim da função é a parte responsável por criar um arquivo externo que recebe as informações inseridas anteriormente como login e senha do usuário e as salva. Cada tentativa poderá ser lida posteriormente no arquivo criado e é apresentada no formato abaixo.

Login: **“login inserido pelo usuário”**– Senha:**“senha inserida pelo usuário”**//[1]

```
 95 #Programa principal
 96
 97 #Abrir tela
 98 root.geometry("600x500")
 99 root.title("Centro Acadêmico de Matemática")
100 root.resizable(0, 0) #Edita o tamanho da tela
101 root.configure()
102
103 #Iniciar o programa
104 tela_inicial()
105 entradas= [] #lista para salvar os logins
106 entrar = tk.Button(root, text="Entar", fg="blue", command= login)
107 entrar.pack()
108 root.mainloop()
```

Esse é o programa principal, onde chamamos as funções que criamos anteriormente. A primeira parte é a responsável pela criação da tela na qual as funções tela_inicial e segunda_tela executarão seus comandos. É chamado o comando root com as funções geometry (determina o tamanho da tela), title(dá o título da tela), resizable (define a mudança ou não do tamanho da tela) e configure (define se o que está impresso na tela pode ser alterado ou não). A segunda parte é o programa. A função tela_inicial é chamada, a lista de logins é criada com o nome entradas (essa lista não foi utilizada nem apagada), o botão que aparecerá na tela principal é criado com o comando que chama a função login e o comando mainloop é adicionado. Esse último comando é o responsável pela possibilidade de se rodar o programa mais de uma vez sem a necessidade de executá-lo novamente.[1]

[1] As linhas do programa que não aparecem nas imagens acima são linhas sem código inseridas com o intuito de separar as partes do programa para facilitar a leitura.

4 CONCLUSÃO

Durante o processo de criação do programa enfrentamos diversos problemas. A maior dificuldade que encontramos foi no uso das bibliotecas do Python. Como não conseguimos baixá-las em nossos computadores pessoais, todo o programa foi feito nos laboratórios da faculdade.

Acreditamos que o processo de criação desse programa nos fez crescer em relação aos conteúdos da matéria uma vez que utilizamos bibliotecas com as quais nunca havíamos tido contato.Abrir telas, fazer com que o programa lesse e criasse novos arquivos e criar botões e caixas de texto foram algumas das novas habilidades que adquirimos ao longo do processo.

Entre altos e baixos, mudamos diversas vezes os caminhos pelos quais iríamos apresentar nosso programa e mesmo com toda a persistência não conseguimos fazer tudo que gostaríamos. A confirmação da senha inserida pelo usuário não deu certo e não conseguimos fazer com que o programa salvasse as alterações feitas pelo usuário administrador.

Entre todas as coisas que nos propusemos a tentar, essas foram as únicas que não conseguimos aprender e aprimorar a tempo de entregar nosso programa. Por esse motivo, achamos que superamos nossas dificuldades e tivemos um saldo positivo em relação a coisas que queríamos e coisas que conseguimos apresentar.

Portanto, apesar de todo o estresse e trabalho que a criação e o desenvolvimento desse programa nos trouxeram, achamos que adquirimos conhecimentos úteis e que poderão um dia nos trazer grande benefício.

5 REFERÊNCIAS

AGUIAR, Vinicius. Seus primeiros passos como Data Scientist: Introdução ao Pandas!. Disponível em: <https://medium.com/data-hackers/uma-introdu%C3%A7%C3%A3o-simples-ao-pandas-1e15eea37fa1>.

The Tkinter Button Widget. Disponível em:<http://effbot.org/tkinterbook/button.htm>.

The TkinterEntryWidget. Disponível em: <http://effbot.org/tkinterbook/entry.htm>.

CAESAR CIPHER. Disponível em: <https://inventwithpython.com/chapter14.html>.

Tkinter. Disponível em: <https://docs.python.org/3/library/tkinter.html>. and<https://docs.python.org/3/library/tkinter.html#module-tkinter>.

The TkinterLabelWidget. Disponível em:<http://effbot.org/tkinterbook/label.htm>.

LIMA, Jackson Prado. Fundamentos. Disponível em: <https://notebooks.azure.com/jacksonpradolima/libraries/programacaopython/tree/Fundamentos>.

Trabalho apresentado pelo professor Jackson Antonio do Prado Lima em sala de aula com nome “Classifica Candidato”