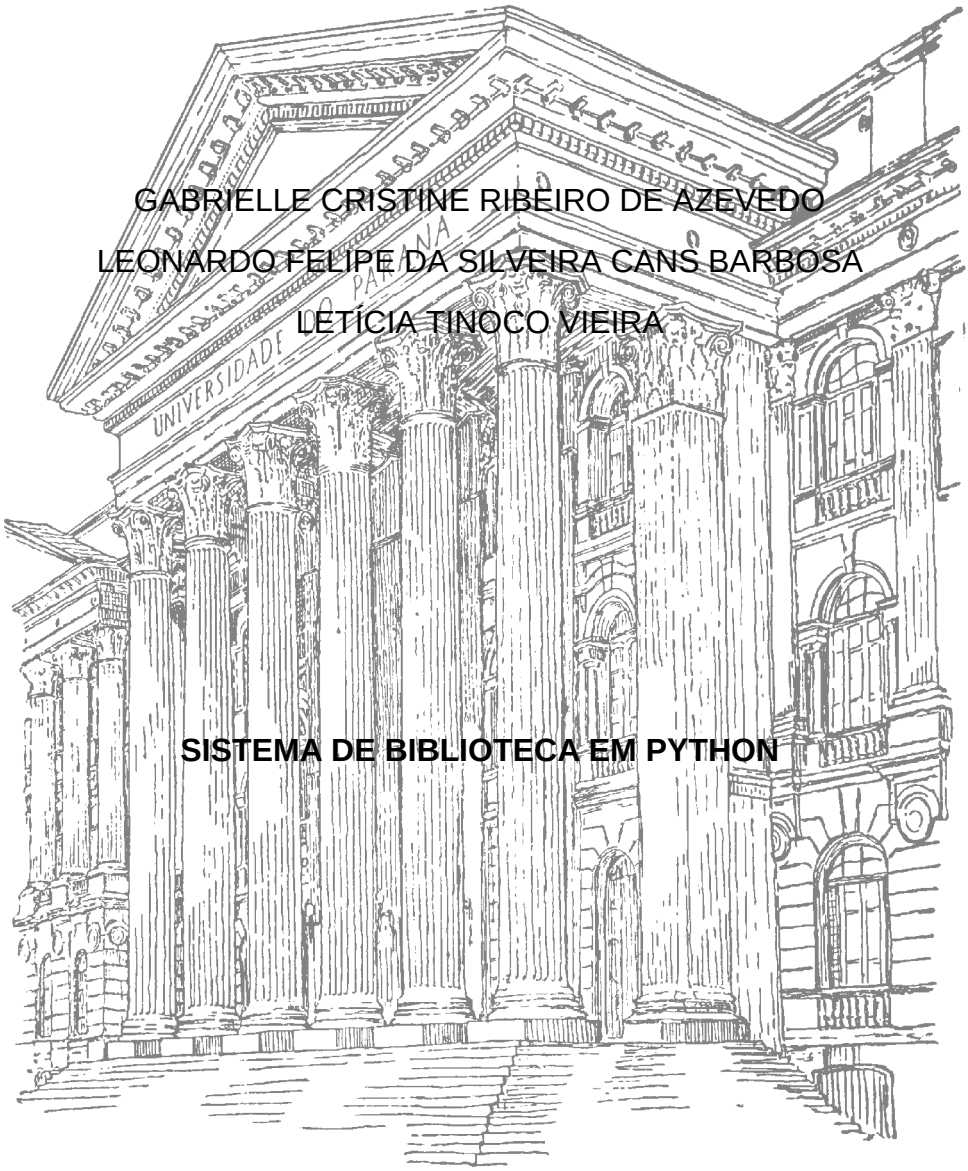
UNIVERSIDADE FEDERAL DO PARANÁ

CURSO DE AGRONOMIA



GABRIELLE CRISTINE RIBEIRO DE AZEVEDO

LEONARDO FELIPE DA SILVEIRA CANS BARBOSA

LETÍCIA TINOCO VIEIRA

SISTEMA DE BIBLIOTECA EM PYTHON

CURITIBA

2018

GABRIELLE CRISTINE RIBEIRO DE AZEVEDO (GRR 20186290)

LEONARDO FELIPE DA SILVEIRA CANS BARBOSA (GRR20180799)

LETÍCIA TINOCO VIEIRA (GRR20187234)

SISTEMA DE BIBLIOTECA EM PYTHON

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

Orientador: Prof. Jackson Antônio do Prado Lima

CURITIBA, Novembro de 2018

SUMÁRIO

1 INTRODUÇÃO

O presente relatório apresenta informações relativas ao trabalho realizado em *Python*. O projeto foi elaborado a fim de criar um sistema de biblioteca que controla o empréstimo e devolução de obras existentes em um acervo.

Python é uma linguagem de programação criada por Guido van Rossum em 1991. Essa linguagem pode ser traduzida por produtividade e legibilidade. *Python* é uma linguagem com um código bom e fácil de manter de maneira rápida, eficiente e simples. Python é dinâmico, multiplataforma, multi-paradigma (orientação à objetos, funcional, refletiva e imperativa), podendo rodar em JVM e .NET Framework. É uma linguagem livre, disponível para programar para desktops, web e mobile.

2 OBJETIVOS

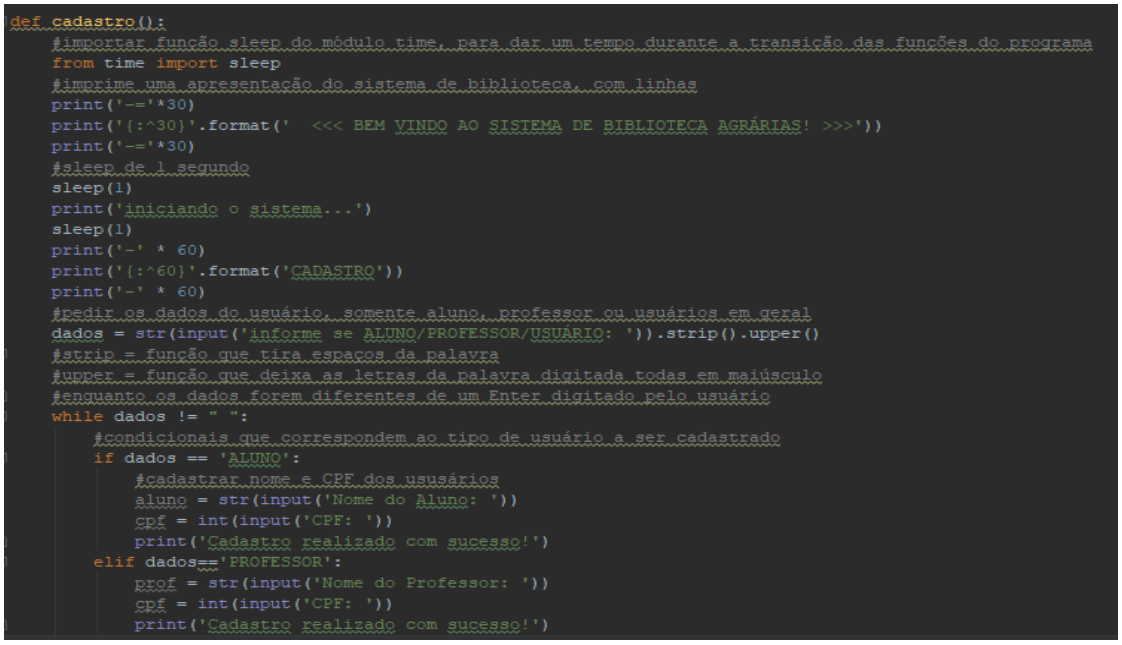
2.1 OBJETIVO GERAL

O objetivo é obter um sistema de biblioteca que possa realizar a pesquisa, cadastro de usuários, controlar o empréstimo e devolução de livros e periódicos criados e cadastrados em dicionários.

3 DESENVOLVIMENTO

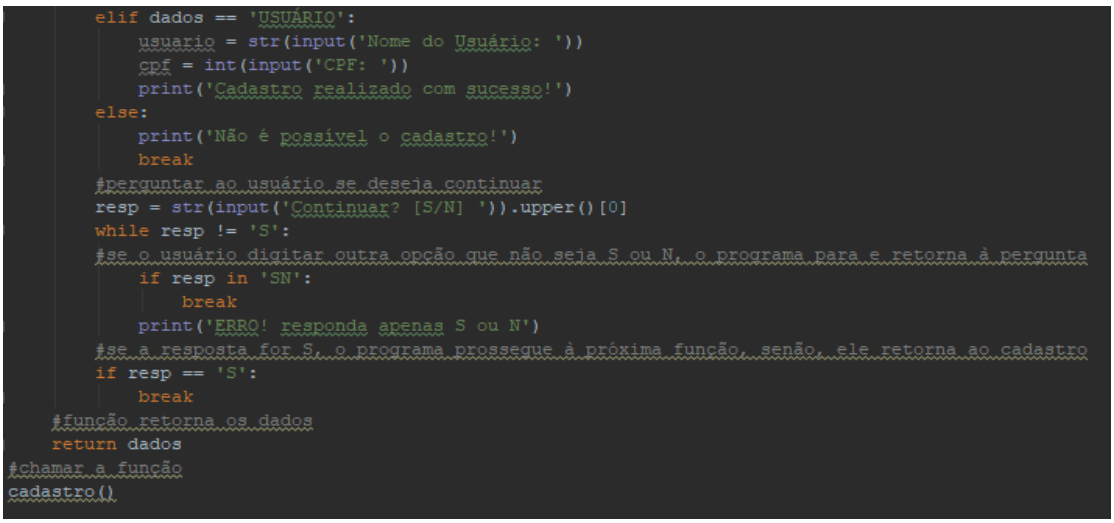
3.1 O TRABALHO

O programa possui 6 funções que serão executadas pela *main*. A primeira função denominada CADASTRO insere os dados pessoais dos usuários da biblioteca, por meio de laço de repetição *while* e condicionais *if, elif, else*. Os usuários poderão ser alunos, professores ou usuários em geral. Os dados pessoais são: nome e CPF. Assim, foi obtido o seguinte código:



```
def cadastro():
    #importar função sleep do módulo time, para dar um tempo durante a transição das funções do programa
    from time import sleep
    #imprime uma apresentação do sistema de biblioteca, com linhas
    print('-='*30)
    print('{:^30}'.format('  <<< BEM VINDO AO SISTEMA DE BIBLIOTECA AGRÁRIAS! >>>'))
    print('-='*30)
    #sleep de 1 segundo
    sleep(1)
    print('iniciando o sistema...')
    sleep(1)
    print('-' * 60)
    print('{:^60}'.format('CADASTRO'))
    print('-' * 60)
    #pedir os dados do usuário, somente aluno, professor ou usuários em geral
    dados = str(input('informe se ALUNO/PROFESSOR/USUÁRIO: ')).strip().upper()
    #strip = função que tira espaços da palavra
    #upper = função que deixa as letras da palavra digitada todas em maiúsculo
    #enquanto os dados forem diferentes de um Enter digitado pelo usuário
    while dados != " ":
        #condicionais que correspondem ao tipo de usuário a ser cadastrado
        if dados == 'ALUNO':
            #cadastrar nome e CPF dos ususários
            aluno = str(input('Nome do Aluno: '))
            cpf = int(input('CPF: '))
            print('Cadastro realizado com sucesso!')
        elif dados=='PROFESSOR':
            prof = str(input('Nome do Professor: '))
            cpf = int(input('CPF: '))
            print('Cadastro realizado com sucesso!')
```

Figura 1.a – Função para cadastro de usuários da biblioteca



```
        elif dados == 'USUÁRIO':
            usuario = str(input('Nome do Usuário: '))
            cpf = int(input('CPF: '))
            print('Cadastro realizado com sucesso!')
        else:
            print('Não é possível o cadastro!')
            break
        #perguntar ao usuário se deseja continuar
        resp = str(input('Continuar? [S/N] ')).upper()[0]
        while resp != 'S':
        #se o usuário digitar outra opção que não seja S ou N, o programa para e retorna à pergunta
            if resp in 'SN':
                break
            print('ERRO! responda apenas S ou N')
        #se a resposta for S, o programa prossegue à próxima função, senão, ele retorna ao cadastro
        if resp == 'S':
            break
    #função retorna os dados
    return dados
#chamar a função
cadastro()
```

Figura 1.b – Função para cadastro de usuários da biblioteca

A segunda função denominada PESQUISA pede ao usuário o título e autor do livro, enquanto que periódico, é solicitado apenas o nome. Para essa função, é necessária a manipulação de dicionários dentro de lista, que são as bibliotecas de livros e periódicos. Para contar a totalidade de exemplares, utiliza-se a função *len*.

Entretanto, nem todas as obras estão disponíveis para empréstimo, logo, deve-se realizar a subtração do total de obras, se livros ou periódicos, pelos exemplares que não são permitidos os empréstimos. Para a pesquisa, foi utilizado laço de repetição simples, *while*, com condicionais *if, elif, else,* e *for, in*, para percorrer a lista de dicionários. Obteve-se o código a seguir:

```
def pesquisa():
    #importa do módulo time a função sleep
    from time import sleep
    #o programa espera 1 segundo para prosseguir com o texto na tela
    sleep(1)
    print('-' * 60)
    print('{:^60}'.format('PESQUISA'))
    print('-' * 60)
    #declaração das variáveis
    resp=" "
    #função len que conta o total de livros e periódicos existentes na biblioteca
    total_livros = len(biblioteca_livros)
    total_periodicos = len(periódicos)
    #foi estabelecido que alguns livros e periódicos não são permitidos para empréstimo
    não_emprestal = 3
    não_emprestap = 3

    #condicionais de pesquisa dos exemplares
    #enquanto a resposta não for S a pergunta de continuar a pesquisa
    while resp != 'S':
        #determinar a totalidade de exemplares e a quantidade que não está disponível para empréstimo

        emprest_livros = total_livros - não_emprestal
        emprest_periodicos = total_periodicos - não_emprestap

        #perguntar o tipo de exemplar
        pesquisa = str(input('Informe tipo de exemplar:[LIVRO/PERIÓDICO] ')).strip().upper()

        #se pesquisa for livro:
        if pesquisa == 'LIVRO':
            livro = input('Digite o título do livro: ').strip().upper()
            autorl = str(input('Digite o autor do livro: '))
```

Figura 2.a– Função para pesquisa de livros e periódicos na biblioteca

```
            #percorrer os dicionários ao longo da lista, verificando os livros existentes
            for v in biblioteca_livros:
                print(v)

            print(f'Resultados da pesquisa: {total_livros} livros no total')
            print(f'Resultados da pesquisa: {emprest_livros} livros disponíveis para empréstimo')

        #se pesquisa for periódico:
        elif pesquisa == 'PERIÓDICO':
            periodico = input('Digite o nome do periódico: ').strip().upper()

            #na biblioteca de periódicos, variável contadora p percorre a lista e imprime todos os periódicos
            for p in periódicos:
                print(p)

            print(f'Resultados da pesquisa: {total_periodicos} periódicos no total')
            print(f'Resultados da pesquisa: {emprest_periodicos} periódicos disponíveis para empréstimo')

        #perguntar ao usuário se deseja verificar as condições de empréstimo
        resp = str(input('Verificar condições do empréstimo? [S/N] ')).upper()[0]
        while not resp == 'S':
            #caso responda outra opção, que não seja S ou N, programa pergunta novamente
            if resp in 'SN':
                break
            print('ERRO! responda apenas S ou N')
        #se a resposta for S, programa prossegue para a próxima função
        if resp == 'S':
            break

#chamar a função pesquisa
pesquisa()
```

Figura 2.b– Função para pesquisa de livros e periódicos na biblioteca

A terceira função denominada EMPRÉSTIMO irá controlar o empréstimo de obras realizado pelo usuário. Por meio de *input*, pergunta-se ao usuário a quantidade de livros e periódicos que ele deseja emprestar. Com laços de repetição e condicionais, é possível controlar a quantidade de obras que o aluno, professor ou usuário em geral podem emprestar. Alunos podem emprestar até 2 obras, e o prazo de devolução é de até 15 dias. Professores podem emprestar até 3 obras ao mesmo tempo, sendo o prazo estendido para até 30 dias, enquanto que usuários em geral somente emprestam 1 exemplar, e o prazo, até 7 dias.

Com a função *date*, do módulo *datetime*, do *python*, calcula-se a data de devolução dos exemplares e a quantidade de dias que o sistema estabelece para entrega de livros e periódicos. Na prática, os dicionários e listas, assim como matrizes, são encontrados em arquivos. Mas para uma melhor didática, utilizou-se lista com dicionários dentro do próprio programa *python*.

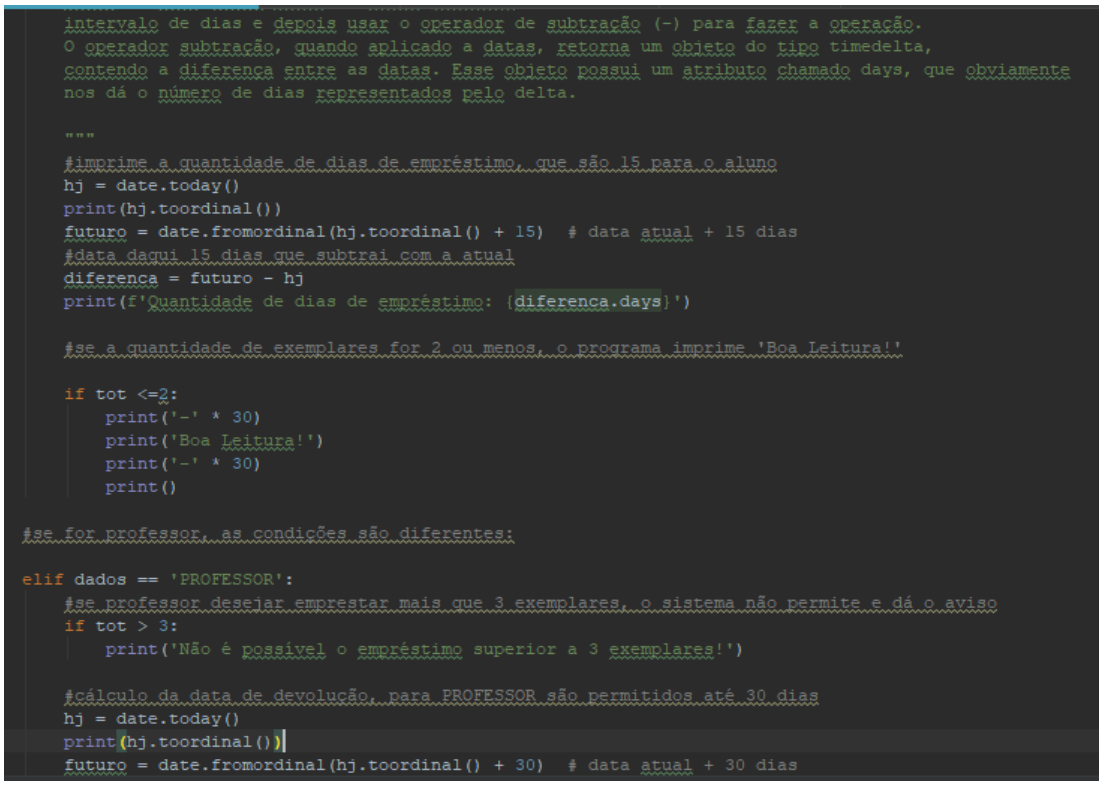
Segue o código da terceira função:

```python
def emprestimo():
    #importar do módulo time, a função sleep
    from time import sleep
    #o programa aguarda 1 segundo para mostrar texto na tela
    sleep(1)
    print('-' * 60)
    print('{:^60}'.format('EMPRÉSTIMO'))
    print('-' * 60)

    #dentro do módulo datetime, importar a função date, que calcula data e hora em python
    from datetime import date
    from datetime import datetime

    #declaração das variáveis
    futuro=0
    resp=" "

    #perguntar tipo de usuário
    dados = str(input('Informe se ALUNO/PROFESSOR/USUÁRIO: ')).upper()
    emp = str(input('Deseja realizar o empréstimo? [S/N] '))
    #após a pesquisa de exemplares, deverá haver o controle do empréstimo
    livro = int(input('Quantidade de livros a serem emprestados: '))
    per = int(input('Quantidade de periódicos a serem emprestados: '))

    #enquanto a resposta a pergunta for diferente de N
    while emp != 'N':
        #total de exemplares emprestados = livros + periódicos
        tot = livro + per
        print(f'Total de exemplares emprestados: {tot} ')

        #há condições de empréstimo para cada tipo de usuário:
```

Figura 3.a– Função de controle de empréstimo de exemplares

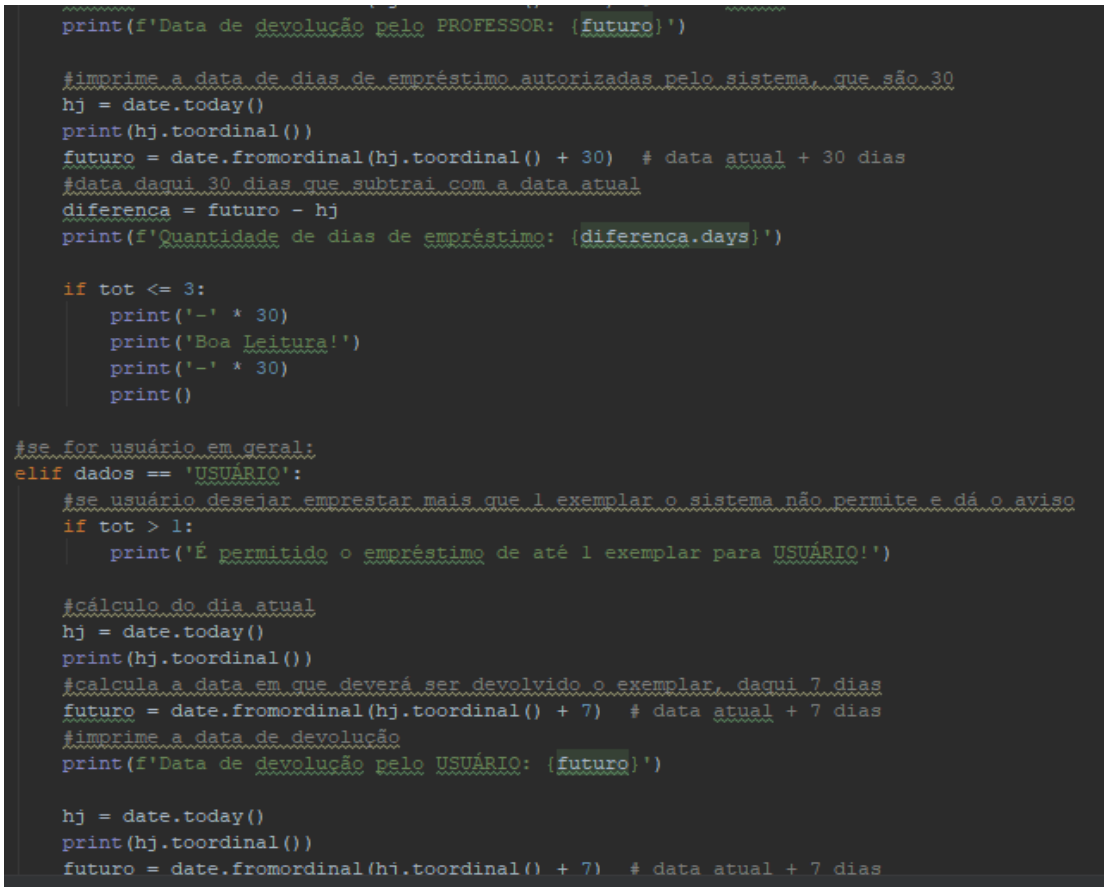
```python
#há condições de empréstimo para cada tipo de usuário:

if dados == 'ALUNO':
#se o aluno desejar emprestar mais que 2 exemplares, o sistema não permite e dá o aviso
    if tot > 2:
        print('Não é possível o empréstimo superior a 2 exemplares para ALUNO!')
    #cálculo de dias de empréstimo
    #são permitidos até 15 dias de empréstimo para ALUNO
    #data atual: dia em que ocorre o empréstimo
    '''
    Data atual:
    '''

    """
    Converter primeiramente a nossa data em um número ordinal, através do método toordinal(),
    que nos retorna a quantidade de dias passados desde o dia 1/1/1 até a data recebida como argumento.
    Depois disso, basta somar (ou subtrair) a esse número inteiro o número de dias da diferença
    que queremos calcular e então reconverter o inteiro para data, através do método fromordinal().
    Abaixo, obtivemos a data a daqui exatos x dias.
    """
    #imprime a data de devolução pelo aluno, que é a data atual + 15 dias de empréstimo possíveis
    hj = date.today()
    print(hj.toordinal())
    #calcula a data de devolução
    futuro = date.fromordinal(hj.toordinal() + 15)  # data atual + 15 dias
    print(f'Data de devolução pelo ALUNO: {futuro}')

    '''
    Calcular a diferença de dias entre datas:
    '''

    """
    Obter as duas datas entre as quais queremos saber o
```

Figura 3.b– Função de controle de empréstimo de exemplares



```
    intervalo de dias e depois usar o operador de subtração (-) para fazer a operação.
    O operador subtração, quando aplicado a datas, retorna um objeto do tipo timedelta,
    contendo a diferença entre as datas. Esse objeto possui um atributo chamado days, que obviamente
    nos dá o número de dias representados pelo delta.

    """
    #imprime a quantidade de dias de empréstimo, que são 15 para o aluno
    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 15)  # data atual + 15 dias
    #data daqui 15 dias que subtrai com a atual
    diferenca = futuro - hj
    print(f'Quantidade de dias de empréstimo: {diferenca.days}')

    #se a quantidade de exemplares for 2 ou menos, o programa imprime 'Boa Leitura!'

    if tot <=2:
        print('-' * 30)
        print('Boa Leitura!')
        print('-' * 30)
        print()

#se for professor, as condições são diferentes:

elif dados == 'PROFESSOR':
    #se professor desejar emprestar mais que 3 exemplares, o sistema não permite e dá o aviso
    if tot > 3:
        print('Não é possivel o empréstimo superior a 3 exemplares!')

    #cálculo da data de devolução, para PROFESSOR são permitidos até 30 dias
    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 30)  # data atual + 30 dias
```

Figura 3.c– Função de controle de empréstimo de exemplares



```
    print(f'Data de devolução pelo PROFESSOR: {futuro}')

    #imprime a data de dias de empréstimo autorizadas pelo sistema, que são 30
    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 30)  # data atual + 30 dias
    #data daqui 30 dias que subtrai com a data atual
    diferenca = futuro - hj
    print(f'Quantidade de dias de empréstimo: {diferenca.days}')

    if tot <= 3:
        print('-' * 30)
        print('Boa Leitura!')
        print('-' * 30)
        print()

#se for usuário em geral:
elif dados == 'USUÁRIO':
    #se usuário desejar emprestar mais que 1 exemplar o sistema não permite e dá o aviso
    if tot > 1:
        print('É permitido o empréstimo de até 1 exemplar para USUÁRIO!')

    #cálculo do dia atual
    hj = date.today()
    print(hj.toordinal())
    #calcula a data em que deverá ser devolvido o exemplar, daqui 7 dias
    futuro = date.fromordinal(hj.toordinal() + 7)  # data atual + 7 dias
    #imprime a data de devolução
    print(f'Data de devolução pelo USUÁRIO: {futuro}')

    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 7)  # data atual + 7 dias
```

Figura 3.d– Função de controle de empréstimo de exemplares

```
            #cálculo da quantidade de dias de empréstimo
            diferenca = futuro - hj
            print(f'Quantidade de dias de empréstimo: {diferenca.days}')

            if tot == 1:
                print('-'*30)
                print('Boa Leitura!')
                print('-' * 30)
                print()

        #pergunta ao usuário se deseja acessar o sistema de devolução
        resp = str(input('Acessar sistema de devolução? [S/N] ')).upper()[0]
        while resp != 'S':
            if resp in 'SN':
                break
            print('ERRO! responda apenas S ou N')
        #se a resposta for S, o programa prossegue para a próxima função
        if resp == 'S':
            break

#chama a função
emprestimo()
```

Figura 3.e– Função de controle de empréstimo de exemplares

```
livro1={'Título': 'Programa de melhoramento da mandioca','Autor': 'Perez e Mendonça', 'Ano':'1973',
 'Editora': 'Sul Publicações'},

livro2={'Título':'Projeto Mandioca','Autor': 'Shermann', 'Ano':'1975',
'Editora': 'Agroeco'},

livro3={'Título':'Proposta do Sistema de Gestão da Formação Profissional do Engenheiro Agrônomo da Universidade Federal '
'do Paraná','Autor': 'Silva e Lemos','Editora': 'UFPR', 'Ano':'2011'},

livro4={'Título':'Recomendação de adubação e calagem no RS e SC: passado, presente e futuro : anais',
'Autor': 'André Lopes, Souza Nunes', 'Editora': 'Morales', 'Ano':'1994'},

livro5={'Título':'Resultados de pesquisa apresentados na XII Reunião da Comissão Sulbrasileira de Pesquisa da Aveia : [anais]',
'Autor': 'João Lourenço', 'Editora': 'Manhatan', 'Ano':'1992'},

livro6={'Título':'Resultados de pesquisa em aveia obtidos na Faculdade de Agronomia da Universidade de Passo Fundo, em 1978',
'Autor': 'Cláudio Santos, Mariana Gonçalves', 'Editora': 'Faculdade de Agronomia da Universidade de Passo Fundo',
 'Ano':'1979'},

livro7={'Título':'Resultados de pesquisas da Fundação Faculdade de Agronomia "Luiz Meneghel", 1970-1984 : resumos',
'Autor': 'Luiz Meneghel','Editora': 'Sul Publicações','Ano': '1986'},

livro8={'Título':'Rotação de culturas','Autor': 'Paolo Azevedo e Moura, Pedro Carlos Lorenzi','Editora': 'Norte',
'Ano':'1968'},

livro9={'Título':'Sanidade e produtividade em búfalos','Autor': 'Estefania Kruschovsky', 'Editora': 'Santana',
'Ano':'1993'}

biblioteca_livros = [livro1, livro2, livro3, livro4, livro5, livro6, livro7, livro8, livro9,]
```

Figura 3.f– Dicionários dentro de lista : Biblioteca de Livros.

```
periódicos = [{'Nome':'Acta scientiarum : agronomy', 'Ano':'2003'},
{'Nome':'Advances in agronomy','Ano':'1949'},
{'Nome':'Agro-Ciencia','Ano':'2000'},
{'Nome':'Agrociência (Montecillo)','Ano':'2015'},
{'Nome':'Agrociencia (Montevideo)','Ano':'1950'},
{'Nome':'Agronomia','Ano':'1941'},
{'Nome':'Agronomia colombiana','Ano':'1983'},
{'Nome':'Agronomia costarricence','Ano':'1977'},
{'Nome':'Agronomia lusitana','Ano':'1939'},
{'Nome':'Agronomia mocambicana','Ano':'1967'}]
```

Figura 3.g– Dicionários dentro de lista : Biblioteca de Livros.

A quarta função é denominada DEVOLUÇÃO. Ela irá controlar a devolução dos exemplares emprestados. Por meio de laços de repetição e usando a função *date*, o programa pergunta se a quantidade de dias de empréstimo foi ultrapassada, e se a resposta for “S” (sim), pergunta-se quantos foram os dias de atraso. Se a quantidade de dias for maior que zero, a multa é calculada, sendo 5 reais por dia de atraso para todos os tipos de usuários. Consequentemente, não é possível o empréstimo de exemplares. São utilizados *while* dentro de *while*. Segue o código da quarta função:

```
def devolução():
    from time import sleep
    sleep(1)
    print('-' * 60)
    print('{:^60}'.format('DEVOLUÇÃO'))
    print('-' * 60)

    #importa do módulo datetime, a função date
    from datetime import date
    #importa do módulo datetime, a função datetime, que calcula data e hora
    from datetime import datetime

    dados = str(input('Informe se ALUNO/PROFESSOR/USUÁRIO: ')).upper()
    #programa pede ao usuário a data em que foi devolvido o exemplar emprestado
    dev = input('Insira a data de devolução dos exemplares: ')

    #enquanto a data de devolução for diferente de um Enter digitado pelo usuário
    while dev != " ":

        #condicionais da devolução
        #se for aluno:
        if dados == 'ALUNO':

            #calcula e imprime a data que o sistema estipulou para a devolução das obras
            hj = date.today()
            print(hj.toordinal())
            futuro = date.fromordinal(hj.toordinal() + 15)
            print(f'Data prevista pelo sistema para devolução dos exemplares pelo ALUNO: {futuro}')

            #calcula e imprime a quantidade de dias que o sistema estipulou para a devolução das obras
            hj = date.today()
            print(hj.toordinal())
```

Figura 4.a – Função de controle de devolução de exemplares

```
print(hj.toordinal())
futuro = date.fromordinal(hj.toordinal() + 15)
diferenca = futuro - hj
print(f'Quantidade de dias de empréstimo prevista pelo sistema: {diferenca.days}')

#pergunta se a quantidade de dias de empréstimo foi ultrapassada
dias = str(input('Quantidade de dias de empréstimo superior ao estipulado pelo sistema? [S/N] ')).upper()[0]
#[0] verifica somente a primeira letra digitada, sendo S ou N

#pergunta a quantidade de dias de atraso da devolução
quant = int(input('Número de dias de atraso: '))

#se a resposta for S, deverá ser aplicada multa pelo atraso da devolução das obras
while dias == 'S':
    if quant > 0:
        #se a quantidade de dias for maior que zero, a multa é calculada
        #multa = 5 reais por dia de atraso
        multa = 5 * quant
        #imprime na tela os dias de atraso;
        print(f'Dias de atraso:{quant}')
        #o valor da multa (2 casas decimais, ou flutuantes);
        print(f'Multa por atraso! Pagamento de R${multa:.2f}')
        #o sistema não permite emprestar exemplares se houve multa por atraso
        print('Não é possível o empréstimo de mais exemplares!')
        #decrementar variável quant, para não haver loop infinito
        quant -= diferenca.days
    break
    #programa para
    #pergunta ao usuário se deseja acessar o menu do sistema

menu = str(input('Acessar menu do sistema: [S/N]')).upper()[0]

if menu == 'S':
```

Figura 4.b – Função de controle de devolução de exemplares

```
        print('ACESSANDO MENU DO SISTEMA DE BIBLIOTECA. POR FAVOR, AGUARDE!')
#se for professor:
if dados == 'PROFESSOR':
    #segue os mesmos processos para aluno

    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 30)
    print(f'Data prevista pelo sistema para devolução dos exemplares pelo PROFESSOR: {futuro}')

    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 30)
    diferenca = futuro - hj
    print(f'Quantidade de dias de empréstimo prevista pelo sistema: {diferenca.days}')

    dias = str(input('Quantidade de dias de empréstimo superior ao estipulado pelo sistema? [S/N] ')).upper()[0]
    quant = int(input('Número de dias de atraso: '))
    #se a quantidade de dias de atraso for superior a zero, deverá ser calculado o valor da multa por atraso
    while dias == 'S':
        if quant > 0:
            multa = 5 * quant
            print(f'Dias de atraso:{quant}')
            print(f'Multa por atraso! Pagamento de R${multa:.2f}')
            print('Não é possível o empréstimo de mais exemplares!')
            quant -= diferenca.days
        break

    menu = str(input('Acessar menu do sistema: [S/N]')).upper()[0]

    if menu == 'S':
        print('ACESSANDO MENU DO SISTEMA DE BIBLIOTECA. POR FAVOR, AGUARDE!')
```

Figura 4.c – Função de controle de devolução de exemplares

```python
#se for usuário
if dados == 'USUÁRIO':

    #segue os mesmos processos do aluno e professor
    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 7)
    print(f'Data prevista pelo sistema para devolução dos exemplares pelo USUÁRIO: {futuro}')

    hj = date.today()
    print(hj.toordinal())
    futuro = date.fromordinal(hj.toordinal() + 7)
    diferenca = futuro - hj
    print(f'Quantidade de dias de empréstimo prevista pelo sistema: {diferenca.days}')

    dias = str(input('Quantidade de dias de empréstimo superior ao estipulado pelo sistema? [S/N] ')).upper()[0]
    quant = int(input('Número de dias de atraso: '))

    while dias == 'S':
        if quant > 0:
            multa = 5 * quant
            print(f'Dias de atraso:{quant}')
            print(f'Multa por atraso! Pagamento de R${multa:.2f}')
            print('Não é possível o empréstimo de mais exemplares!')
            quant -= diferenca.days
        break
```

Figura 4.d – Função de controle de devolução de exemplares

```python
            menu = str(input('Acessar menu do sistema: [S/N]')).upper()[0]

            #se a resposta for S, o programa segue para a função menu
            if menu == 'S':
                print('ACESSANDO MENU DO SISTEMA DE BIBLIOTECA. POR FAVOR, AGUARDE!')

        #programa sai do loop existente na função e prossegue para a próxima função, se o usuário desejar
        break

#chama a função devolução
devolução()
```

Figura 4.e – Função de controle de devolução de exemplares

A quinta função é denominada MENU. Nela o usuário digita a opção desejada, retornar a uma das funções anteriores ou sair do programa. Segue o código da quinta função:

```python
def menu():
    from time import sleep
    sleep(0.8)
    '''
    Função que contém o menu do programa
    '''

    # Mostra as opções para o usuário na tela
    print('MENU:')
    print('\t1 - cadastro')
    print('\t2 - pesquisa')
    print('\t3 - emprestimo')
    print('\t4 - devolução')
    print('\t5 - Sair')

    # Retorna o que o usuário digitar
    return int(input('Digite a opção desejada: '))

menu()
```

Figura 5 – Função MENU

A sexta função é denominada MAIN. Ela é o ponto de partida para a execução do programa. Em geral, ela controla a execução direcionando as chamadas para outras funções no programa. Normalmente, um programa para de ser executado no final de *main*, embora possa ser terminado em outros pontos por diversos motivos. Segue o código da sexta função:

```
def main():
    '''
    Programa principal
    '''

    # guardaremos os dados em um vetor ou array
    dados = []
    # chama a função menu que mostra o menu e solicita uma opção ao usuário
    opcao = menu()
    # enquanto o usuário não digitar a opção: 5 - Sair
    while opcao != 5:
        # Se for a opção: 1 - CADASTRO
        if opcao == 1:
            # chama a função que insere dados no vetor de dados
            cadastro()
        # Se for a opção: 2 - PESQUISA
        elif opcao == 2:
            # chama a função que pesquisa livros e periódicos na tela
            # passando o vetor que contém esses dados
            pesquisa()
            time.sleep(2)
        # Se for a opção: 3 - EMPRÉSTIMO
        elif opcao == 3:
            # chama a função que realiza o controle de empréstimo de exemplares:
            emprestimo()
            time.sleep(2)
        # Se for a opção: 4 - DEVOLUÇÃO
        elif opcao == 4:
            # chama a função que realiza o controle de devolução de exemplares:
            devolução()
            time.sleep(2)
```

Figura 6.a – Função MAIN

```
        # Se digitou uma opção de menu inválida
        else:
            print("\nOpção inválida! Por favor, informe uma opção válida.\n")
            # Faz o programa esperar 1 segundo e depois continua
            time.sleep(1)

        # Limpa a tela do terminal para melhorar o aspecto visual da apresentação
        os.system('cls' if os.name == 'nt' else 'clear')

        # Mostra ao usuário o menu novamente e aguarda ele escolher uma opção
        opcao = menu()

    # Salva os dados em um arquivo, antes do término do programa
    salva_dados_arquivo(nome_arquivo, dados)


if __name__ == "__main__":
    '''
    Ao ser executado o programa python, esse trecho de código será executado e o programa principal será chamado.
    '''

    # Chama a função main
    main()
```

Figura 6.b – Função MAIN

Vale ressaltar a utilização dos laços dentro de laços de repetições, em todas as funções, além do uso de *if, elif, else*. É importante o uso das funções dos módulos do *python*: *date*() e *sleep*(), ambas importadas de módulos, *datetime* e *time*, respectivamente Além de funções que auxiliam na eficácia e eficiência do código, além da praticidade ao usuário, tais como *upper*() e *strip*(). Todas elas corroboram para um código mais organizado e apresentável ao usuário.

4 CONCLUSÃO

O trabalho contribuiu para uma maior prática e aprendizado da disciplina de fundamentos de programação de computadores. Por meio do *python,* é possível realizar inúmeras tarefas por meio da criação de códigos de maneira mais simples que outras linguagens de programação. Com a extensa pesquisa e prática de inúmeros exercícios, dos conteúdos estudados, como laços de repetição, condicionais, funções, listas, matrizes, além de estudos adicionais acerca de dicionários e tuplas, foi possível criar o código apresentado nesse relatório. A tarefa de planejar e colocar em prática o que fora aprendido foi muito complexa. Entretanto, esse é o primeiro passo para muitas outras criações, e como em qualquer outra ciência, demanda muito esforço, trabalho e disciplina.

REFERÊNCIAS

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). Informação documentação - Trabalhos acadêmicos - Apresentação. **NBR14724**. Rio de Janeiro, 2011.

AMADEU, M. S. U. et al. **Manual de normalização de documentos científicos de acordo com as normas da ABNT**. Curitiba, 2015.