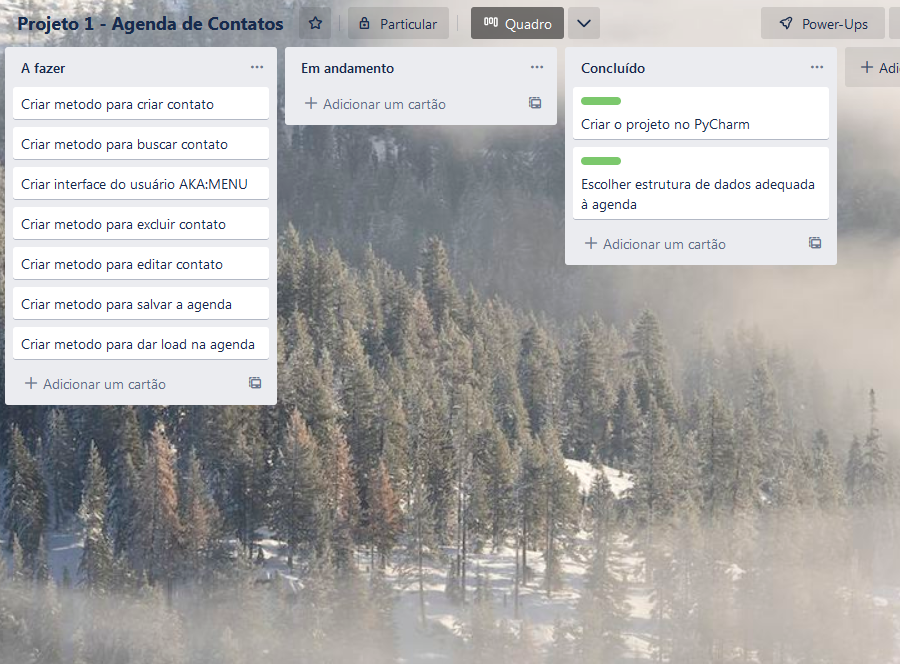
Iniciando projeto – definindo estrutura de dados

#Estrutura de dados em dicionario  
#Criando dicionario dentro de dicionario{} para estruturar os valores  
agenda = {  
 'leo': {  
 'celular': '449987-5210',  
 'email': 'leonardo@email.com',  
 'endereco': 'Rua X'  
 },  
 'ana': {  
 'celular': '449956-4520',  
 'email': 'ana@email.com',  
 'endereco': 'Rua Y'  
 },  
 'joao': {  
 'celular': '449958-6310',  
 'email':'joao@email.com',  
 'endereco':'Rua Z'  
 }  
}

Usando trello para gerenciar tarefas



1 – Metodo mostrar contatos

def mostrar\_contatos(contato = ''):  
 for contato in AGENDA:  
 #vai chamar a funcao buscar contato passando o for, no qual vai printar todos os contato da agenda  
 buscar\_contato(*contato*)  
 print('##################################')

2 – Buscar contato especifico

def buscar\_contato(*contato\_procurado*):  
 try:  
 print('Nome:', *contato\_procurado*)  
 print('Telefone:', AGENDA[*contato\_procurado*]['celular'])  
 print('Email:', AGENDA[*contato\_procurado*]['email'])  
 print('Endereço:', AGENDA[*contato\_procurado*]['endereco'])  
 except:  
 print('Contato não encontrado')

3 – Adicionando contato

def incluir\_contato(*nome*, *celular*, *email*, *endereco*):  
 AGENDA[*nome*] = {'celular': *celular*,  
 'email': *email*,  
 'endereco': *endereco*,  
 }  
 print('>>>>>>Contato {} adicionado com sucesso'.format(*nome*))

4 – Editando contato

def editar\_contato(*nome*, *campo*, *novo\_valor*):  
 AGENDA[*nome*][*campo*] = *novo\_valor* print('>>>>>> {} de {} alterado com sucesso'.format(*campo*, *nome*))

5 – Excluindo contato

def excluir\_contato(*nome*):  
 AGENDA.pop(*nome*)  
 print('>>>>>> {} removido com sucesso'.format(*nome*))

6 – Criando menu com if

def mostrar\_menu():  
 print('1 - Mostrar todos os contatos')  
 print('2 - Buscar contato')  
 print('3 - Adicionar contato')  
 print('4 - Editar contato')  
 print('5 - Excluir contato')  
 print('6 - Fechar agenda')

def selecao\_menu(*opcao*):  
 # mostrando todos contatos  
 if *opcao* == '1':  
 mostrar\_contatos()  
 # buscando contato  
 elif *opcao* == '2':  
 proc = input('Digite o nome do contato: ')  
 buscar\_contato(proc)  
 # adicionando contato  
 elif *opcao* == '3':  
 nome = input('Nome: ')  
 celular = input('Celular: ')  
 email = input('Email: ')  
 endereco = input('Endereço: ')  
 incluir\_contato(nome, celular, email, endereco)  
 # editando contato  
 elif *opcao* == '4':  
 nome = input('Nome do contato: ')  
 campo = input('Campo a ser alterado: [celular], [email], [endereço]')  
 valor = input('Novo valor: ')  
 editar\_contato(nome, campo, valor)  
 # excluindo contato  
 elif *opcao* == '5':  
 nome = input('Nome: ')  
 excluir\_contato(nome)  
 else:  
 print('Opção inválida')

7 - Menu com repetição (while)

opcao = None  
while(opcao != '6'):  
 mostrar\_menu()  
 opcao = input('Escolha uma opção: ')  
 selecao\_menu(opcao)

Enquanto não for escolhida a opção 6 (de saída do programa) a cada loop o menu será executado novamente

**Adicionando tratamento de erros ao programa**

Se contato procurado não existir vai dar erro

def buscar\_contato(*contato\_procurado*):  
 try:  
 print('Nome:', *contato\_procurado*)  
 print('Celular:', AGENDA[*contato\_procurado*]['celular'])  
 print('Email:', AGENDA[*contato\_procurado*]['email'])  
 print('Endereço:', AGENDA[*contato\_procurado*]['endereco'])  
 except KeyError:  
 print('Contato inexistente')  
 except Exception as erro:  
 print('Um erro inesperado ocorreu')  
 print(erro)

Se contato não existir vai dar erro

def excluir\_contato(*nome*):  
 try:  
 AGENDA.pop(*nome*)  
 print('>>>>>> {} removido com sucesso'.format(*nome*))  
 except KeyError:  
 print('Contato inexistente')  
 except Exception as erro:  
 print('Um erro inesperado ocorreu')  
 print(erro)

Se agenda estiver vazia vai dar erro

def mostrar\_contatos():  
 if AGENDA:  
 for contato in AGENDA:  
 #vai chamar a funcao buscar contato passando o for, no qual vai printar todos os contato da agenda  
 buscar\_contato(contato)  
 print('##################################')  
 else:  
 print('Agenda vazia')

Se contato não existir vai dar erro e sair da edição

# editando contato  
elif *opcao* == '4':  
 try:  
 nome = input('Nome do contato: ')  
 AGENDA[nome]  
 print('>>>>>> Editando contato: {}'.format(nome))  
 campo = input('Campo a ser alterado: [celular], [email], [endereço]: ')  
 valor = input('Novo valor: ')  
 editar\_contato(nome, campo, valor)  
 except:  
 print('Contato não existe')

**Extras**

**Abrindo arquivos**

Abrindo arquivo

arquivo = open(‘e-mails.txt’)

Transformando arquivo em lista para printar com método readLines()

conteúdo = arquivo.readLines()

método strip() limpa a string removendo espaços em branco e outros

for linha em conteúdo:

print(linha.strip())

Sempre ao abrir arquivo temos que fecha-lo após parar de usar

Arquivo.close()

Podemos usar o with para evitar erros no código, ao final ele fecha o arquivo sozinho

Try:

with open(‘e-mails.txt’) as arquivo:

arquivo.readLines()

except:

print(‘arquivo nao encontrado’)

**Escrevendo em arquivos**

Try:

Parametros: **r**(read) ler, **w**(write)escreve e cria novo arquivo, **a**(append) adiciona mais coisas no arquivo sem sobrescrever dados existentes)

With open(‘arquivo.txt’, a) as arquivo:

Arquivo.write(‘teste123\n’)

except Exception as erro:

print(‘algum erro ocorreu’)

print(erro)

Exportando agenda para arquivo .csv

def exportar\_contato():  
 try:  
 with open('agenda.txt', 'w') as arquivo:  
 for contato in AGENDA:  
 celular = AGENDA[contato]['celular']  
 email = AGENDA[contato]['email']  
 endereco = AGENDA[contato]['endereco']  
 arquivo.write('{},{},{},{}\n'.format(contato, celular, email, endereco))  
 print('>>>>>>Arquivo exportado com sucesso')  
 except Exception as erro:  
 print('>>>>>>Algum erro ocorreu durante a exportação')  
 print(erro)