

Health Search

Sival Leão de Jesus

Departamento de Ciências Exatas – Universidade Estadual de Feira de Santana (UEFS)
Av. Transnordestina, s/n, Novo Horizonte, 44036-900, Feira de Santana, BA, Brasil

Abstract. *The report will contain the methods used to develop the software, which will aim to optimize the search time, at the request of a large hospital in Feira de Santana, where the Python programming language was used.*

Resumo. *O relatório conterá os métodos utilizados para o desenvolvimento do software, que terá como finalidade otimizar o tempo de busca, a pedido de um grande hospital de Feira de Santana, onde foi utilizado a linguagem de programação Python.*

1. Introdução

A demora nos atendimentos nos hospitais vem crescendo bastante devido a alta demanda de pacientes, a falta de eficiência na hora de encontrar documentos importantes, como relatórios e laudos, proporcionará atrasos no atendimento e nos demais serviços do hospital, ocasionando enormes filas.

Com o intuito de organizar os documentos de um grande hospital em Feira de Santana, que está enfrentando problemas de organização, os alunos da Universidade Estadual de Feira de Santana, foram convocados a desenvolver um software na linguagem de programação python, para solucionar este problema, onde o programa deve auxiliar na gestão dos documentos hospitalares.

O programa deve realizar ações básicas, como receber os diretórios dos usuários e guarda no software onde está armazenado os arquivos para quando o usuário pesquisa não perder muito tempo, utilizando este serviço o mesmo poderá agilizar a busca e o próprio atendimento do hospital.

Diante a dificuldades expostas este relatório tem como finalidade, esclarecer e explicar a construção do software, capaz de realizar a busca dos documentos para agilizar e organizar os procedimentos hospitalares.

2. Metodologia

Durante as sessões tutoriais foi discutido qual seria a melhor forma de buscar os arquivos .txt e guardá-los de maneira que os dados não fossem perdidos quando encerrasse o programa. Como neste problema o uso de arquivos é fundamental, utilizamos deste mesmo artifício para salvar os dados, gerando arquivos .txt de salvamento com dados que podem ser interpretado pelo software como uma linha de código, onde se transformava uma estrutura do python chamada de dicionário para arquivo .txt e convertido novamente como um código pela função eval que interpreta automaticamente o que está escrito e transforma em linha de código.

Antes de se iniciar o processo de desenvolvimento do código foi feito testes para entender melhor o funcionamento de ler e escrever arquivos .txt utilizando linguagem de programação python, depois desta etapa foi se iniciado o processo de desenvolvimento do software responsável por ler todos arquivos no formato .txt presente no diretório fornecido pelo usuário para que assim facilite a busca e aumente a eficiência nos procedimentos, o usuário também tem a liberdade de remover diretórios contendo arquivos ou atualizá-lo e verificar quais arquivos estão registrados.

2.1. Processo de construção

O software foi desenvolvido no sistema operacional Windows, desenvolvido pela Microsoft, na linguagem de programação Python. Inicialmente foi feito o “import os“, funções prontas sobre o sistema operacional que não são nativas do Python mas que são de extrema importância para o funcionamento do software, com a implementação do import foram possíveis localizar os arquivos presente nos diretórios. Com os dados necessários obtidos pelo import era possível localizar e ler os arquivos utilizando a função “with open()” onde se era retornando uma “string”, o próprio texto do arquivo, e logo depois esse texto era separado por palavras e inseridos em dicionário, estrutura da linguagem python para armazenar dados, porém assim que fechasse o software esses dados seriam perdidos, com intuito de solucionar esse impasse foi feito a conversão deste dicionário em arquivo .txt assim sempre que o programa for iniciado é feito a leitura e transformado novamente em dicionário pela função “eval()” que avalia automaticamente o que é fornecido e transforma em uma expressão em python.

2.2. Definição de requisito

Foram solicitado vários requisitos que o software deve atender, entre eles estão, ler os arquivos no formato .txt, atualizá los caso o mesmo diretório fosse inserido novamente,

remoção de um diretório ou um arquivo específico, busca por uma palavra-chave onde o resultado deve aparecer em ordem decrescente, exibir todos os índices já presente no código e ajuda caso o usuário digitar um comando inválido onde o software deveria realizar suas operações em argumento por linha de comando. Requisitos necessários para que os funcionários percam o menor tempo possível e agilizem o trabalho e o atendimento.

2.3. Ler arquivos e atualizar

O processo para ler os arquivos é feito através da função “with open()”, função que possibilita abrir o arquivo e fechá-lo sozinho assim que terminar o processo e inserir todos os dados escritos no arquivo em uma variável para que seja possível manipulá-la normalmente. Assim que a variável recebe o texto as suas palavras são separadas com a função “split()” depois as palavras iguais são contadas e é adicionado a um dicionário contendo a palavra, o caminho que ela está localizada e quantas vezes apareceu no arquivo, esse processo é feito até que todos os arquivos .txt presente no diretório fornecido seja lido, contado e adicionado no dicionário.

Para atualização do arquivo é verificado se o diretório inserido já havia passado pelo processo de leitura e em seguida todos os dados deste diretório eram removidos do dicionário, palavras, endereço do arquivo e número de vezes que a palavra se repetiu. Depois do processo de remoção de todos os dados, o diretório fornecido pelo usuário voltava à seção de leitura como se fosse a primeira vez.

2.4. Remoção

O processo de remoção deveriam ocorrer em duas situações com o usuário inserindo um diretório ou arquivo específico porém o software desenvolvido só consegue remover o diretório completo e não somente os dados de um arquivo específico.

Para a remoção de um diretório primeiramente foi feito uma laço de repetição “for” para que pegasse todos os arquivos presente no diretório depois foi utilizando a função “pop()” do python para remover do dicionário onde essa função vai receber as chaves, argumento definido para separar os item do dicionário, para que assim só apague o que realmente está presente naquele diretório, mantendo os demais documentos inalterados, as palavras que só havia naquele diretório tem seus dados completamente apagados evitando que fique qualquer dado vazio podendo interferir no processo de busca.

2.5. Busca

O processo de busca contava com um requisito específico onde era necessário mostrar os arquivos que contêm a palavra da busca na ordem decrescente avaliando a quantidade de vezes que a palavra aparece no arquivo, foi feita a tentativa de ordenar utilizando o método de ordenação “selection sort” porém não foi efetivo, mostrando assim totalmente fora de ordem.

Para o processo de busca foi feito um dicionário onde contém uma lista com todas as palavras separada por arquivo, esse mesmo dicionário foi usado para verificar se a palavra existe em algum arquivo para que não ocorresse erros na hora de fazer a busca, depois da verificação através de um laço de repetição “for” o software procura os termos semelhantes ao digitado e os separa em uma lista contendo o nome do arquivo, também são separadas em outras lista as demais informações necessária como, o diretório que o arquivo está localizado e a quantidade de vezes que o termo apareceu no arquivo, todos esses dados foi posto em lista diferente, e como todos foram posto no em uma lista na mesma ordem eles têm o mesmo índice porém estão em listas distintas, os espaços que os dados são armazenado em lista são chamados de índices, que são representados por número começando do 0 (zero) e cada elemento possui o seu separadamente, e para mostrar esses dados sem misturar ou trocar foi feito um loop de repetição com o “for” com a quantidade total de itens e com o auxílio de uma variavel que recebe o numero dos indices da lista e era mostrado na tela os dados índice por índice de tres listas simultaneamente.

2.6. Exibir dados cadastrados

Semelhante a busca foi utilizadas listas para separar os dados necessários para ser mostrado como o nome do arquivo e o local que ele está localizado no computador do usuário. E esses dados foram retirados de um dicionário só que foi acessado de elemento a elemento e posto em duas lista separado o nome do arquivo e seu endereço segundo a mesma lógica vista anteriormente em busca, pois foi julgado ser melhor desta maneira.

2.7 Ajudar o usuário

O software desenvolvido tem somente uma entrada principal, onde o usuário insere um comando para realizar as ações do programa, com comando já definidos onde as operações são realizadas através de condições e se nenhuma destas condições for atendidas é mostrado na tela todos os comando e suas funções para que assim o usuário

saiba quais são os comandos existentes e o que cada comando realiza, como é mostrada na imagem 1.

```
digite um comando: Comando inválido
-----
                                COMANDOS DO SOFTWARE
-----

FUNÇÕES
-d = adicionar diretório ou atualizar
-b = buscar arquivo por termo
-r = remover diretório
-m = mostrar arquivos indexado
-----
```

Imagem 1. Mensagem de ajuda

3. Manual de uso

Ao iniciar o software aparecerá uma mensagem escrita “digite um comando” onde o usuário deverá escrever o comando da ação que ele deseja realizar onde entre eles estão as letras “d” para adicionar diretório ou atualizar, “b” para buscar arquivo por termo, “r” para remover diretório e “m” para mostrar todos arquivos indexado, para realizar as ações descritas todas as letras devem ser precedidas do sinal de menos “-”. Exemplos -d, -b, -r, e -m, para o funcionamento correto.

3.1. -d

Após o “-d” ser inserido será necessário que o usuário informe o diretório que ele deseja que os arquivos sejam lidos, se o diretório inserido não existir o software é fechado sem fazer nenhum cadastro caso contrário ele é fechado mas depois de armazenar todos os dados. Apenas os arquivos do diretório inseridos serão guardados, os arquivos das subpastas serão desconsiderados.

3.2. -b

Após o “-b” ser inserido será necessário que o usuário informe o termo que deseja buscar com distinção de letras maiúsculas e minúsculas, também deve se respeitar os sinais de pontuação das palavras. Depois do termo ser inserido corretamente é mostrado na tela o nome do arquivo que contém aquele termo, o endereço onde está localizado o arquivo e a quantidade que o termo aparece no arquivo, se for inserido um termo inexistente aparecerá uma mensagem dizendo “Termo não encontrado” e logo em seguida o programa é fechado.

3.3. -r

Após o “-r” ser inserido será necessário que o usuário informe o diretório que ele deseja que os arquivos sejam removidos, se o diretório inserido não existir o software é fechado sem remover nenhum dado, caso contrário ele é fechado mas depois de remover todos os dados. Apenas os arquivos do diretório inseridos serão apagados, os arquivos das subpastas serão desconsiderados.

3.4. -m

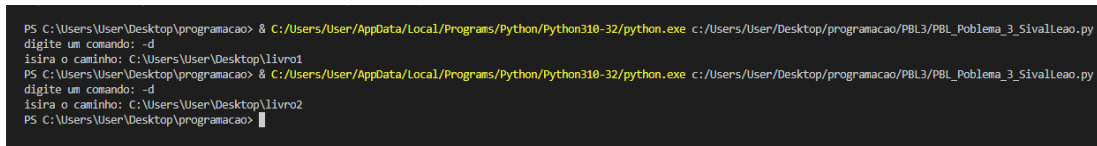
Logo após o “-m” ser inserido é mostrado na tela o nome e o endereço de todos os arquivos guardados no software.

3.5. Mensagem de ajuda

Se for inserido qualquer valor diferente dos vistos acima apareceram os comandos -d, -b, -r, e -m, e suas funções para orientar o usuário.

4. testes

Para as realizações dos testes foram inseridos dois diretórios contendo um arquivo .txt em cada um, como mostra a imagem 2.



```
PS C:\Users\User\Desktop\programacao> & C:/Users/User/AppData/Local/Programs/Python/Python310-32/python.exe c:/Users/User/Desktop/programacao/PBL3/PBL_Problema_3_Sivalleao.py
digite um comando: -d
insira o caminho: C:\Users\User\Desktop\livro1
PS C:\Users\User\Desktop\programacao> & C:/Users/User/AppData/Local/Programs/Python/Python310-32/python.exe c:/Users/User/Desktop/programacao/PBL3/PBL_Problema_3_Sivalleao.py
digite um comando: -d
insira o caminho: C:\Users\User\Desktop\livro2
PS C:\Users\User\Desktop\programacao> █
```

Imagem2. Inserindo diretório

e três arquivos de textos foram criados no computador do usuário, entre eles o “save.txt”, “save2.txt” e o “save3.txt” responsáveis por guardarem os dados para o software não perder quando fosse encerrado.

Em seguida foi feita uma busca nos arquivos com o termo educação como é mostrado na imagem 3.

```
digite um comando: -b
digite o termo: educação
-----
NOME: livro1.txt
ENDEREÇO DO ARQUIVO: C:\\Users\\User\\Desktop\\Livro1\\livro1.txt:
QUANTIDADE DE 'educação' PRESENTE: 2
-----
NOME: livro2.txt
ENDEREÇO DO ARQUIVO: C:\\Users\\User\\Desktop\\Livro2\\livro2.txt:
QUANTIDADE DE 'educação' PRESENTE: 29
```

Imagem 3. Realizando a busca.

Logo após foi removido do “Livro1.txt” todas as palavras “educação”, e foi feito a atualização em seguida a busca novamente como pode ser observado na imagem 4.

```
PS C:\Users\User\Desktop\programacao> & C:/Users/User/AppData/Local/Programs/Python/Python310-32/python.exe c:/Users/User/Desktop/programacao/PBL3/PBL_Poblema_3_Sivalleao.py
digite um comando: -d
insira o caminho: C:\Users\User\Desktop\Livro1
PS C:\Users\User\Desktop\programacao> & C:/Users/User/AppData/Local/Programs/Python/Python310-32/python.exe c:/Users/User/Desktop/programacao/PBL3/PBL_Poblema_3_Sivalleao.py
digite um comando: -b
digite o termo: educação
-----
NOME: livro2.txt
ENDEREÇO DO ARQUIVO: C:\\Users\\User\\Desktop\\Livro2\\livro2.txt:
QUANTIDADE DE 'educação' PRESENTE: 29
```

Imagem4. Atualizando o índice.

Depois foi usado a função de mostrar todos os arquivos guardados como mostrado na imagem 5.

```
digite um comando: -m
-----
NOME DO ARQUIVO: livro2.txt
PASTA DO ARQUIVO: C:\Users\User\Desktop\Livro2
-----
NOME DO ARQUIVO: livro1.txt
PASTA DO ARQUIVO: C:\Users\User\Desktop\Livro1
PS C:\Users\User\Desktop\programacao> █
```

Imagem 5. Arquivos guardados.

E por último foi removido o “Livro2.txt” do software e em seguida foi mostrado os arquivos guardados como é mostrado na imagem 6.

```
digite um comando: -r
digite o caminho do arquivo a ser excluído: C:\Users\User\Desktop\Livro2
PS C:\Users\User\Desktop\programacao> & C:/Users/User/AppData/Local/Programs/Python/Python310-32/python.exe c:/Users/User/Desktop/programacao/PBL3/PBL_Poblema_3_Sivalleao.py
digite um comando: -m
-----
NOME DO ARQUIVO: livro1.txt
PASTA DO ARQUIVO: C:\Users\User\Desktop\Livro1
PS C:\Users\User\Desktop\programacao> █
```

Imagem 6. Remoção de diretório.

5. Conclusão

Todos os objetivos propostos foram feitos da melhor maneira possível, porém infelizmente não foram cumpridos completamente, faltando alguns detalhes para armazenar arquivos como a possibilidade de inserir dois arquivos com o mesmo nome e com diretórios diferentes, remoção de um arquivo somente e mostra a busca em ordem decrescente e o principal de todos que não possibilita executar por linha de comando.

5. Referências

Import os - código fonte:

<https://github.com/python/cpython/tree/3.10/Lib/os.py>, Acesso em: 02 de jun. de 2022