Dengue Free Feira

Lucas Oliveira da Silva

Engenharia de Computação – Universidade Estadual de Feira de Santana (UEFS)
Av. Transnordestina, S/N. Feira de Santana, Novo Horizonte – BA, Brasil – 44036-900.

lucasoliveiradaailva@gmail.com

Resumo. *Com o crescimento exponencial dos casos de Dengue, é de suma importância o desenvolvimento de um sistema de gerenciamento dos dados dessa doença. Tendo em vista essa problemática, os alunos da disciplina MI algoritmos, foram solicitados para elaborar um software de gerenciamento dos dados da Dengue em Feira de Santana. Dessa forma, este relatório traz uma breve explicação do processo de elaboração desse sistema: metodologia adotada, dificuldades ao longo da elaboração, bem como os resultados e possíveis melhorias.*

1. Introdução

O Brasil vem sofrendo com um alto índice de casos da Dengue, doença transmitida pelo mosquito *Aedes aegypti*. A Bahia enfrenta um aumento alarmante de casos, segundo dados da Secretaria de Estado da Saúde da Bahia, SESAB, o crescimento ultrapassa em 688% o período anterior. Feira de Santana como a segunda maior cidade do estado, está com um alto número de casos, e com isso a quantidade de dados cresce exponencialmente.

1.1 Resumo do problema

Nesse sentido, os alunos da disciplina MI Algoritmos da Universidade Estadual de Feira de Santana, UEFS, receberam uma solicitação da vigilância sanitária feirense para desenvolver, individualmente, um sistema para gerenciar os dados da Dengue na cidade. O software deveria atender os seguintes requisitos básicos presentes na tabela 1.

Tabela 1. Requisitos básicos do sistema.

Requisitos básicos do sistema
Ler dados de um arquivo csv
Realizar operações com os dados
Mostrar os dados de forma clara
Escrever os dados no arquivo
Mostrar informações acerca da doença e sistema

1.2 Descrição breve da solução

O sistema desenvolvido na linguagem Python, foi projetado em módulos, sendo cada um responsável por uma tarefa específica. Ele tem como base as seguintes estruturas:

- Dicionários e listas para armazenar e processar todos os dados;
- Condicionais e Estruturas de repetição para gerenciar o fluxo do sistema e operar os dados de forma automática;
- Funções para permite uma reutilização posterior do código, além de Variáveis e outras estruturas básicas;

Ao iniciar o sistema, o usuário vai escolher entre três opções: gerenciar dados, mais informações e sair do sistema. "Gerenciar dados" permite ao usuário ler e escrever dados, já o "mais informações" permite visualizar informações da Doença e do software. A opção "sair do sistema" finaliza a execução e salvar os dados no arquivo CSV.

2. Metodologia

Durante as sessões tutoriais a equipe levantou como primeira questão, "como o Python ia armazenar os dados do arquivo?", para isso, optei por utilizar a biblioteca CSV que é nativa da linguagem. Ademais, foi necessário um estudo acerca das estruturas vetores e dicionários. Em seguida, pensamos em armazenar os dados em dicionários ou matrizes, optei pela primeira estrutura, tendo em vista que seria mais "fácil" de manipular. Além disso, desenvolvi a interface no modelo de uma tabela. No contexto de escolhas coletivas, ficou determinado que a mensagem presente na opção "mais informações" seria padrão para todos da equipe.

Por fim, desenvolvi esse sistema utilizando a programação modular, isto é, pegar uma lógica grande e ir transformando em algo relativamente simples e pequeno. Para isso separei o sistema em quatro módulos: main, operações dados, leitura de dados e salvar dados.

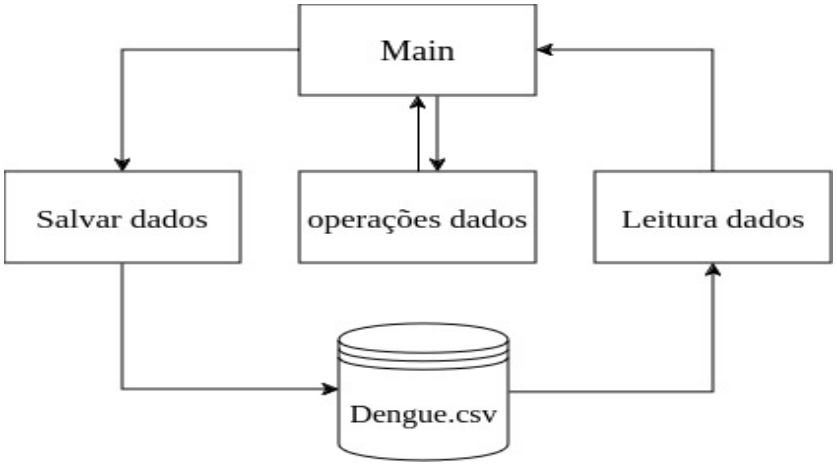

Figura 1. Estrutura do projeto.

2.1 Definição de requisitos

Para atender os requisitos básicos presentes na tabela 1, foi necessário seguir algumas determinações:

1. Desenvolver um menu para conseguir ligar as opções dos requisitos básicos do sistema;

2. Para atender ao 1º requisito, seria necessário desenvolver a leitura de dados de um arquivo csv com os casos da dengue. Atrelado a isso, para atender 4º requisito, o arquivo deveria possibilitar a atualizações, trabalhando sempre com dados atualizados;

3. Já para atender o 3º requisito, o sistema deveria mostrar o total de casos em uma data, bairro e toda feira. Além disso, ao escolher um bairro, o sistema mostraria a porcentagem dos casos desse bairro em relação a soma dos outros bairros de feira. Por fim, deveria permitir a comparação percentual para duas datas de um mesmo bairro, mostrando se houve aumento ou diminuição dos casos.

2.2 Descrição de alto nível

Para desenvolver esse sistema, primeiramente foi necessário elaborar o módulo de leitura de dados, tendo em vista que ele seria responsável por puxar os dados do arquivo csv. Esse módulo possui apenas uma função principal “obter_dados_formatados”, ela é sempre chamada por fora, sendo responsável por chamar as funções auxiliares desse módulo e organizar os dados na seguinte hierarquia: feira/data/bairro/dados_bairro. A variável feira é um dicionário que possui como chaves as datas, dentro das datas temos outro dicionário que tem cada bairro como chave, por fim, dentro da chave de cada bairro tem seus respectivos dados.

As funções auxiliares são responsáveis por pequenas tarefas, uma acessa a base de dados, enquanto outra fica responsável pelo empacotamento no dicionário, e no final, retorna os dados empacotados para a função principal do módulo.

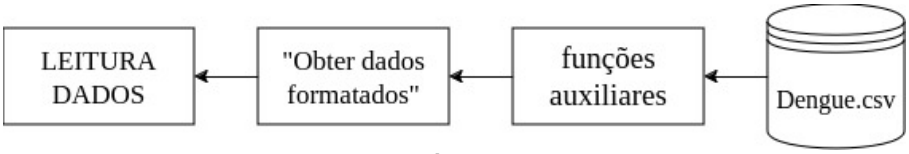

Figura 2. Estrutura do Módulo “Leitura de Dados”.

Depois de ler os dados com o módulo “Leitura dados”, seria necessário que esses dados retornassem para a base, isto é, o arquivo. Para isso, desenvolvi o módulo “Salvar dados”, seu objetivo principal é salvar e desempacotar os dados do dicionário. Basicamente, ele vai pegar aqueles dados, organizar tudo e salvar no arquivo csv. Isso foi feito utilizando uma função principal chamada “salvar_dados”, seguida por outras funções auxiliares. Cabe ressaltar que assim como no módulo “leitura dados” somente uma função é que opera as outras do módulo, neste caso é a função “salvar_dados”.

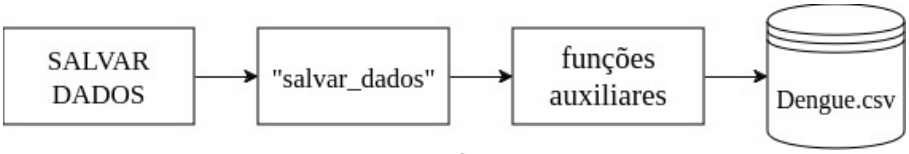

Figura 3. Estrutura do Módulo “Salvar dados”.

Com o sistema lendo e salvando os dados, seria necessário agora as operações com os dados, ou seja, os cálculos, porcentagens, visualização de forma clara e objetiva ect. Tendo em vista essas necessidades que surgiu o módulo “operações dados”, ele é responsável por operar o sistema, funcionando como um “backend”.

Ele possui quatro funções principais: “ler todos os dados”, “ler dados por data”, “ler dados por bairro” e “escrever dados”. Essas funções operam com outras auxiliares, que podem ser chamadas dependendo da execução do sistema. A função “ler todos os dados”, mostra todos os dados do sistema e seus respectivos cálculos. O funcionamento das funções “Ler dados por data” e “Ler dados por bairro” é similar, enquanto uma tem opera com datas outra é com bairros. Por fim, a função “escrever dados” permite que o usuário escreva novos dados no sistema, vale ressaltar que o sistema só aceita datas maiores que a última registrada, para isso foi utilizado a biblioteca “datetime”, nativa do python.

Figura 4. Estrutura do Módulo “Operações dados”.

Por fim, desenvolvi o módulo “main”, ele é quem controla os outros módulos, sendo responsável pela chamada das funções principais de cada módulo. Além das chamadas dos módulos, ele é composto por menus para as opções escolhidas pelo usuário, optei por deixar no main, pois não vi necessidade de criar um módulo só para menus.

Em síntese, o sistema funciona assim: o módulo “leitura de dados” ler e organiza os dados de um arquivo para um dicionário. Esse dicionário chega ao módulo “main” e logo em seguida é enviado para o módulo “operações dados”, dependendo da escolha do usuário, ele faz a respectiva operação. Ao sair do sistema, os dados do dicionário são enviados para o módulo “salvar dados”, que passa do dicionário para o arquivo csv.

2.3 Ordem de codificação

O processo de desenvolvimento desse sistema durou cerca de 5 semanas. Na primeira semana realizei os estudos acerca da estrutura do projeto: dicionários, listas, tuplas e a programação modular.

Já na segunda semana, comecei o desenvolvimento na prática, estudei a documentação da biblioteca “csv” do python. Desses estudo surgiu o primeiro algoritmo do módulo leitura de dados, sua função é pegar os dados do arquivo e converter para a

hierarquia do dicionário. Nessa mesma semana, desenvolvi o módulo “salvar dados”, com seu algoritmo principal que seria converter e salvar o dicionário no arquivo csv, no entanto, o sistema estava com um erro na hora de passar os dados no arquivo, estava colocando os dados por bairro.

Na terceira semana resolvi o erro, apenas mudei a hierarquia do dicionário, passando de “feira/bairro/data/dados” para “feira/data/bairro/dados”. Além disso, utilizei a biblioteca “datetime” para ordenar as datas de forma crescente. O motivo da escolha dessa biblioteca, foi porque o usuário poderia atualizar diretamente do arquivo, e isso poderia quebrar o sistema, logo essa ferramente se mostrou essencial.

Na quarta semana, desenvolvi os menus do módulo “main”, conectando os outros módulos a ele, já que o mesmo seria o responsável pela execução do sistema. Além disso, comecei a desenvolver o módulo “operações dados”, nessa semana apenas implementei somente a leitura dos dados, sem nenhum cálculo. Infelizmente, surgiu um grande erro na visualização dos dados, desenvolvi uma forma de pintar o terminal com códigos de escape, os dados estavam sendo visualizados como a figura 5, entretanto quando o sistema era executado em uma IDE (Integrated Development Environment), ocorria um erro nas cores que foram programadas, figura 6, por isso resolvi remover as cores da tabela.

Bairro:	Data:	Habitantes:	Casos Suspeitos:	Casos Negativos:	Casos Confirmados:
Tomba	22/03/2024	55007	100	20	500
Campo Limpo	22/03/2024	47060	30	40	120
Muchila	22/03/2024	22496	30	10	66
Conceição	22/03/2024	21694	30	20	55
Brasília	22/03/2024	21168	400	200	89
Mangabeira	22/03/2024	20819	55	100	26
Calumbi	22/03/2024	19858	66	3	42
Queimadinha	22/03/2024	19203	99	5	98
Gabriela	22/03/2024	17618	20	8	2
Parque Ipê	22/03/2024	16469	50	10	4
Jardim Cruzeiro	22/03/2024	14694	57	5	5
Rua Nova	22/03/2024	13078	9	8	9
Lagoa Grande	22/03/2024	12229	8	9	87
Aviário	22/03/2024	11912	55	7	46
Santa Mônica	22/03/2024	11617	20	4	26
Centro	22/03/2024	11382	15	6	98
Pedra de Descanso	22/03/2024	11156	5	2	42
Caseb	22/03/2024	10982	9	99	2
São João	22/03/2024	10239	7	88	15
Cidade Nova	22/03/2024	9974	10	200	9
Jardim Acácia	22/03/2024	9009	2	30	8
Serraria Brasil	22/03/2024	8368	5	10	44
Baraúna	22/03/2024	8093	99	9	21
Cis	22/03/2024	7887	55	5	25
Ponto Central	22/03/2024	7221	20	10	85
Tomba	23/03/2024	55007	90	25	505
Campo Limpo	23/03/2024	47060	50	40	130
Muchila	23/03/2024	22496	43	12	70
Conceição	23/03/2024	21694	20	30	55
Brasília	23/03/2024	21168	350	210	1096
Mangabeira	23/03/2024	20819	50	100	36
Calumbi	23/03/2024	19858	70	5	44
Queimadinha	23/03/2024	19203	111	21	108
Gabriela	23/03/2024	17618	30	21	16

Figura 5. Sistema executado no terminal Linux.

airro:	Data:	Habitantes:	Casos Suspeitos:	Casos Negativos:	Casos Confir
omba	22/03/2024	55007	100	20	500
ampo Limpo	22/03/2024	47060	30	40	120
uchila	22/03/2024	22496	30	10	66
onceição	22/03/2024	21694	30	20	55
rasília	22/03/2024	21168	400	200	89
angabeira	22/03/2024	20819	55	100	26
alumbi	22/03/2024	19858	66	3	42
ueimadinha	22/03/2024	19203	99	5	98
abriela	22/03/2024	17618	20	8	2
arque Ipê	22/03/2024	16469	50	10	4
ardim Cruzeiro	22/03/2024	14694	57	5	5
ua Nova	22/03/2024	13078	9	8	9
agoa Grande	22/03/2024	12229	8	9	87
viário	22/03/2024	11912	55	7	46
anta Mônica	22/03/2024	11617	20	4	26
entro	22/03/2024	11382	15	6	98
edra de Descanso	22/03/2024	11156	5	2	42
aseb	22/03/2024	10982	9	99	2
ão João	22/03/2024	10239	7	88	15
idade Nova	22/03/2024	9974	10	200	9
ardim Acácia	22/03/2024	9009	2	30	8
erraria Brasil	22/03/2024	8368	5	10	44
araúna	22/03/2024	8093	99	9	21
is	22/03/2024	7887	55	5	25
onto Central	22/03/2024	7221	20	10	85
omba	23/03/2024	55007	90	25	505
ampo Limpo	23/03/2024	47060	50	40	130
uchila	23/03/2024	22496	43	12	70
onceição	23/03/2024	21694	20	30	55
rasília	23/03/2024	21168	350	210	1096
angabeira	23/03/2024	20819	50	100	36
alumbi	23/03/2024	19858	70	5	44

Figura 6. Sistema executado na IDE Visual Studio Code.

Na quinta semana, desenvolvi, de fato, o módulo “operações dados”, implementei funções para realizar todos os cálculos, porcentagens e visualização de dados. Esse é o maior módulo do sistema, ele é o “backend”. Desenvolvi a escrita de dados no sistema, e mais uma vez, fiz uso da biblioteca “datetime”, dessa vez para verificar se a data inserida é maior que a última data do sistema. Além disso, refatorei algumas funções e por fim documentei cada função e módulo para permitir uma fácil leitura.

O sistema foi desenvolvido na IDE Visual Studio Code, com o interpretador Python 3.10.12, no sistema operacional Linux utilizando a distro Zorin OS. Além disso, como exposto anteriormente, o software foi desenvolvido utilizando as bibliotecas csv e datetime, ambas nativas da linguagem Python..

3. Resultados e Discussões

Após as cinco semanas de desenvolvimento, o sistema ficou pronto e funcionando perfeitamente, seguindo a estrutura projetada e aberto a futuras melhorias.

Para utilizar o Dengue Free Feira, é necessário ter um interpretador do Python na versão mais recente. O usuário deve executar o arquivo com nome “main”. Ele é o módulo principal do sistema, O menu inicial do sistema oferece três opções enumeradas: gerenciar dados, mais informações e sair.

Caso o usuário queira trabalhar com os dados, deve selecionar Gerenciar dados, que abrirá um menu contendo as opções: Ler dados, Escrever dados e retornar ao menu anterior.

A opção Ler dados, abre um menu para o usuário escolher como visualizar os dados, podendo escolher as opções: todos os dados, dados por data, dados por bairro e retornar ao menu anterior. Todos os dados mostra os casos em forma de tabela, a soma total de casos e a porcentagem em relação aos casos notificados. A opção dados por data, tem as mesmas funcionalidades da anterior, com a diferença da escolha de uma data específica, após a escolha, o sistema irá mostrar todos os dados referentes a essa data. Por fim, a opção dados por bairro, pede ao usuário o nome de bairro e mostra seus dados, além do percentual em relação a feira, logo em seguida, o sistema pergunta ao usuário, se ele quer comparar os dados de duas datas desse bairro.

Retornando ao menu gerenciar dados e entrando na opção escrever dados, o sistema pede ao usuário, que digite os dados seguindo a seguinte estrutura: Bairro|Data|Habitantes|Casos Suspeitos|Casos Negativos|Casos Confirmados. Cabe ressaltar que é necessário respeitar essa ordem, para que não ocorra um erro na distribuição dos dados. Além disso, a data inserida tem que ser maior ou igual a última data registrada no sistema, caso seja igual, é necessário que os dados para o bairro escolhido ainda não tenham sido registrados.

Por fim, retornando ao menu principal e acessando a opção mais informações, o sistema informa sobre a doença dengue e sobre a finalidade do sistema. Para finalizar a execução do sistema e salvar os dados, o usuário deve escolher a opção sair. Assim os dados são enviados para a base de dados, isto é, o arquivo csv.

3.1 Dados de entrada

O sistema não possui uma validação de dados para as entradas do usuário, pois consideramos que todas os dados iram ser digitados corretamente. Para operar entre os menus e selecionar alguma opção, digite o número inteiro referente aquele opção, exemplo: [1] ler dados.

Para escolher datas é importante seguir a estrutura “dd/mm/yyyy”, inserindo todos os número no formato inteiro. Por fim, como explicado anteriormente para escrever dados é necessário seguir o modelo: Bairro|Data|Habitantes|Casos Suspeitos|Casos Negativos|Casos Confirmados. O nome do bairro precisa iniciar em maiúsculo e ser do formatado String (texto), o formato da data segue do mesmo modelo explicado anteriormente, e os dados restantes precisam ser números inteiros, no total serão passados 6 dados, é importante que todos sejam fornecidos ao sistema.

3.2 Testes

Ao longo do desenvolvimento do sistema, realizei diversos testes nas funções para garantir o funcionamento adequado. Por fim, fiz o teste principal do sistema adicionando os novos dados da figura 7.

Bairro:	Data:	Habitantes:	Casos Suspeitos:	Casos Negativos:	Casos Confirmados:
Tomba	24/03/2024	55007	100	20	500
Campo Limpo	24/03/2024	47060	30	40	120
Muchila	24/03/2024	22496	30	10	66
Conceição	24/03/2024	21694	30	20	55
Cidade Nova	24/03/2024	9974	10	200	9
Centro	25/03/2024	11382	17	7	105
Pedra de Descanso	25/03/2024	11156	6	3	45
Caseb	25/03/2024	10982	10	105	3
São João	25/03/2024	10239	8	95	18
Cidade Nova	25/03/2024	9974	11	220	12
Jardim Acácia	25/03/2024	9009	3	35	9
Serraria Brasil	25/03/2024	8368	6	12	48
Baraúna	25/03/2024	8093	105	10	23
Cis	25/03/2024	7887	60	6	28
Ponto Central	25/03/2024	7221	22	12	90

Figura 7. Dados de teste.

A opção todos os dados mostra os seguintes totais: habitantes = 419233, casos suspeitos = 1300, casos Negativos = 1045, casos confirmados = 2655. Enquanto a porcentagem em relação ao total notificado foi: casos suspeitos 26%, casos confirmados 53.1% e casos negativos 20.9%.

Enquanto escolhendo a data “25/03/2024” na opção dados por data, o sistema mostra: habitantes = 94311, casos suspeitos = 248, casos Negativos = 505, casos confirmados = 381. Já selecionando “tomba” em dados por bairro, o sistema retorna os dados da ultima data desse bairro, no caso dia “24/03/2024” que é : habitantes = 55007 casos suspeitos = 100, casos negativos = 20 e casos confirmados = 500. Ainda nessa opção, o bairro Tomba representa 7,69% dos casos Suspeitos totais. Além disso, ele representa 18,83% dos casos confirmados totais.

3.3 Erros

O sistema foi projetado visando entradas corretas, logo, ele não verificar o tipo de dado inserido, por isso, é fundamental que o usuário leia o tópico de dados de entrada. Além

disso, identifiquei que quando o usuário não insere nenhum dado, dentro da opção de escrever dados, o sistema lança um erro e fecha. Isso ocorre porque o sistema considera que os dados foram inseridos e tenta fazer o empacotamento nesses dados inexistentes.

4. Conclusão

Em síntese, o sistema foi concluído com sucesso, ele segue a estrutura modular projetada, sendo cada módulo responsável por uma determinada tarefa. Ele alcançou todos os requisitos básicos e específicos. A possível melhoria é a refatoração de algumas linhas, tendo em vista que poderiam ser mais claras e coesas. Além disso, o desenvolvimento de uma interface para o usuário tonaria o sistema mais elegante. Em resumo, os resultados foram positivos e o sistema atende o seu propósito.

5. Referências

Bahia tem 265 municípios em epidemia de Dengue . Saúde BA, 2024. Disponível em: <https://www.saude.ba.gov.br/2024/04/08/bahia-tem-265-municipios-em-epidemia-de-dengue/> Acesso em 4 de junho de 2024.

WAZLAWICK, R. S. (2018). Introdução a Algoritmos e Programação com Python. Elsevier.

MANZANO, J. A. N. G.; OLIVEIRA, J. F. (1996). Algoritmos: Lógica para Desenvolvimento de Programação. Érica,

Documentação Python. Disponível em: <https://docs.python.org/3/> Acesso em 27 de abril de 2024.

MAYNARD, Jeff, 1976. Programação modular. LTC.