Sistema de Gerenciamento de Doações e Montagem de Cestas Básicas

Amanda Lima Bezerra

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

limaa9000@gmail.com

Resumo. O Dispensário Santana solicitou aos alunos de engenharia da computação um programa capaz de contabilizar e organizar cestas básicas para doação. O programa foi feito em Python no sistema operacional Windows 11 dentro da IDE PyCharm. Com o propósito de esclarecer toda a metodologia usada na construção do programa e seus erros de funcionamento foi feito este relatório.

1. Introdução

Fundado oficialmente em 29 de maio de 1983 [História do dispensário Santana, 2010] o Dispensário Santana é uma instituição sem fins lucrativos que tem como propósito ajudar a comunidade mais carente de Feira de Santana. Mas a história da instituição começou há alguns anos, no ano de 1946, com o monsenhor Amílcar Marques de Oliveira, nono pároco de Sant'Ana, juntamente com um grupo de idosas que arrecadaram alguns donativos para doação [História do Dispensário Santana, 2010]. O grupo acabou por ser chamado de Dispensário Santana.

As necessidades da comunidade iam além de doações de alimentos e roupas, por isso ao longo dos anos a área de atuação da instituição foi se expandindo. Atualmente o Dispensário Santana oferece, além de doações, creche, posto de saúde, asilo, cursos, entre outros [Dispensário Santana: 36 anos transformando vidas, 2019]. Todas essas ações só são possíveis graças às parcerias feitas ao longo dos anos e às doações, o que foi o objetivo inicial da fundação. Visando facilitar a organização e cadastro de doações, o Dispensário Santana contactou os alunos do curso de Engenharia de Computação da Universidade Estadual de Feira de Santana para desenvolver um sistema em Python que possa facilitar a organização das cestas básicas.

O sistema proposto foi feito na linguagem de programação Python, usando funções e estruturas básicas, visto que o problema não demandava um sistema mais rebuscado. Ademais, o programa solicitado deve contabilizar as doações de alimentos e ao final apresentar quantas cestas básicas podem ser formadas e, caso exista, doações excedentes. Ao longo deste relatório será descrito mais detalhes a respeito do funcionamento, construção e problemas do programa, por hora a descrição ficará a respeito da linguagem de programação usada.

Python tem sua origem em 1989 a partir de uma necessidade do seu criador Guido Van Rossum por uma linguagem de sintaxe semelhante à da linguagem ABC, mas que tivesse acesso ao sistema Amoeba [Guilherme Lima, 2021]. É uma linguagem de alto nível, ou seja, sua escrita se aproxima muito da linguagem falada, por isso tem se tornado muito popular.

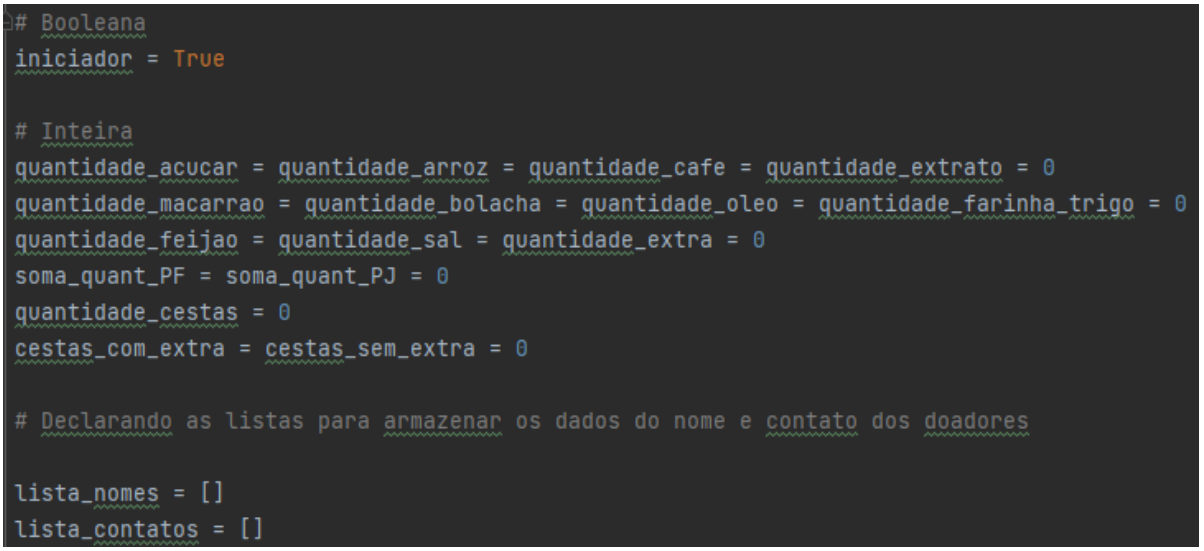
2. Metodologia

O programa foi discutido e desenvolvido ao longo das sessões da aula do MI – Algoritmos, espaço onde foi direcionado como e de qual forma o programa seria construído. Além disto, decisões individuais também fizeram parte da construção desse projeto. O conhecimento usado para a construção desse programa foi obtido através de estudos prévios, visto que poucas coisas foram aprendidas no decorrer da execução. Inicialmente esse tópico descreverá desde as funções e estruturas usadas até o funcionamento, do ponto de vista do usuário e do programador, do sistema.

O objetivo principal do programa é cadastrar doações de alimentos e ao final mostrar quantas cestas básicas poderão ser formadas. Cumprido o objetivo principal, o sistema deve contabilizar quantas cestas terão itens extra ou não e quantas doações foram feitas por cada tipo de doador, físico ou jurídico, e ao final apresentar a quantidade de itens restantes.

Do ponto de vista do usuário o programa inicialmente solicita o nome e contato do doador, visto que isso foi exigido no problema, e em seguida é perguntado qual tipo de doador, podendo ser físico ou jurídico. Assim que todos esses dados são preenchidos corretamente o menu de doações é apresentado. Lembrando que todas as entradas, excetuando nome e contado do doador, estão protegidas de erros. O menu, o objetivo principal do programa, é o local onde serão cadastradas todas as doações. Ao final o usuário terá a opção de sair do menu, onde irá decidir se continua o cadastro de outro doador ou encerra o programa. Caso a escolha seja não cadastrar mais um doador, o sistema automaticamente apresentará os resultados.

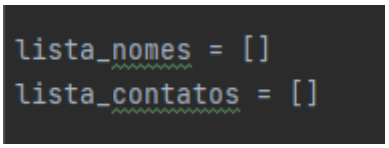
Do ponto de vista do código, o programa começa com uma declaração de variáveis que precisam de um valor inicial, seja ela *True or False* ou um inteiro (Figura 1). Por uma questão de funcionalidade do código algumas variáveis não foram declaradas nesse ponto a exemplo de iniciador_itens e quantidade, linhas 58 e 95 respectivamente. A primeira, caso declarada nas primeiras linhas do projeto, não iria fazer a repetição do menu no momento correto, visto que caso algum doador fosse doar mais de um tipo de item iria ter que se cadastrar novamente. A segunda ocasionaria uma contagem errada dos itens doados por pessoas físicas ou jurídicas, já que o último valor dado a variável quantidade iria ser contabilizada duas vezes antes de sair do *loop* do menu.



```
# Booleana
iniciador = True

# Inteira
quantidade_acucar = quantidade_arroz = quantidade_cafe = quantidade_extrato = 0
quantidade_macarrao = quantidade_bolacha = quantidade_oleo = quantidade_farinha_trigo = 0
quantidade_feijao = quantidade_sal = quantidade_extra = 0
soma_quant_PF = soma_quant_PJ = 0
quantidade_cestas = 0
cestas_com_extra = cestas_sem_extra = 0

# Declarando as listas para armazenar os dados do nome e contato dos doadores

lista_nomes = []
lista_contatos = []
```

Figura 1

Uma decisão individual tomada para este problema foi adicionar duas listas para o registro dos nomes e contatos dos doadores, estas também foram declaradas no início do código vazias (Figura 2). O programa apresenta uma mensagem de boas-vindas e em seguida entra no *loop* principal, que abrange a maior parte do programa. A função desse w*hile* é justamente coletar todos os dados necessários. O *loop* secundário abarca somente a parte do cadastro das doações, contabilidade do total de produtos recebidos e somatório das doações das pessoas físicas e jurídicas. Ao sair de todos os loops existentes se entra no tratamento de dados, sejam eles contabilizados ou calculados. A seguir será discutido mais a fundo tópicos importantes para o entendimento de como o programa foi pensado e em como ele funciona.

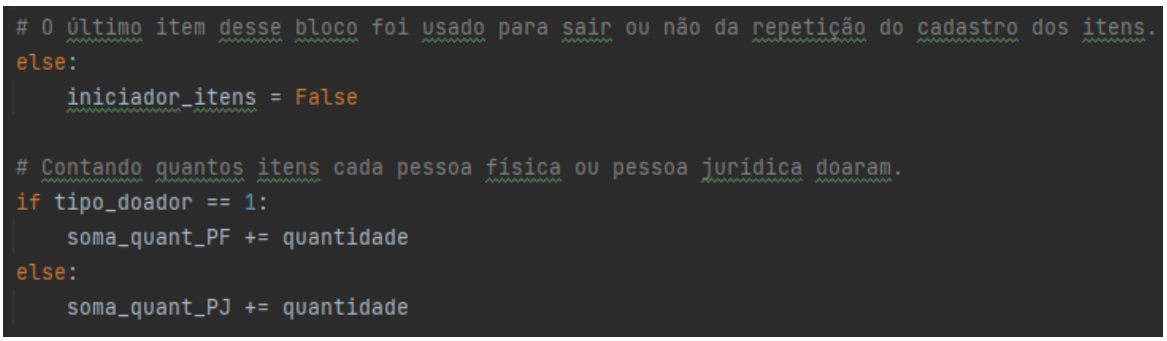


```
lista_nomes = []
lista_contatos = []
```

Figura 2

2.1 Estruturas condicionais e de repetição

Estruturas condicionais e de repetição foram usadas ao longo do projeto todo, visto que são estruturas fundamentais para o funcionamento de qualquer programa e é notável a presença de diversos *if* , *elif* ou *else* e *while* no código. Os blocos *if, elif* e *else* trabalham em conjunto, pois quando uma das condições é atendida automaticamente não se entra nas outras condições. Mas no momento em que se tem diversos *ifs* o caminhar natural do programa é verificar um por um. Por uma questão de otimização e facilitar a leitura do código, todas as estruturas condicionais que estão relacionadas devem ser escritas primeiramente com *if* seguido por quantos *elifs* necessitar e por último um *else.* No programa há um trecho onde existem duas condicionais *if* (Figura 3) mas que são independentes uma da outra, visto que elas estão verificando condições diferentes tem objetivos diferentes no código.



```
# O último item desse bloco foi usado para sair ou não da repetição do cadastro dos itens.
else:
    iniciador_itens = False

# Contando quantos itens cada pessoa física ou pessoa jurídica doaram.
if tipo_doador == 1:
    soma_quant_PF += quantidade
else:
    soma_quant_PJ += quantidade
```

Figura 3

A única estrutura de repetição usada no código foi o *while*, sendo ela a melhor opção já que não se sabe quantos doadores serão cadastrados. O funcionamento do *while* é baseado em uma condição, se ela for verdadeira o programa entra dentro do bloco, caso contrário não será executado. As variáveis usadas para adentrar o *while* foram iniciador e iniciador_itens, ambas têm valores booleanos de *True* e caso o usuário não necessite mais cadastrar doadores ou mais alimentos ambas irão assumir o valor *False* e consequentemente sair do *while*.

2.2 Funções

As funções, assim como as estruturas condicionais e de repetição, são essenciais no programa. Nesse projeto só foram usadas funções nativas do Python que são o *print*, *input*, *int*, *float*, *isnumeric* e *isinstance*. O *print* nada mais é uma função que mostra na tela o que está escrito dentro dela ou o valor da variável, *input* é usada para receber dados do usuário, *int* e *float* para converter uma *string* em um número inteiro ou em um número de ponto flutuante respectivamente.

Passando por essas funções básicas a função *isnumeric* e *isinstance* são as que demandam uma explicação mais profunda do seu funcionamento. A primeira, função *isnumeric,* é uma função que recebe um único parâmetro e seu funcionamento consiste em uma comparação para saber se é ou não um numeral, para ao final retornar um valor booleano de *True* ou *False.* A última, *isinstance*, é uma função que recebe dois parâmetros, uma variável e uma classe, com o objetivo de retornar também um valor booleano. Seu funcionamento é também um comparativo onde se verifica se a variável pertence ou não a aquela classe. Ambas, funções *isnumeric* e *isinstance*, foram usadas com o propósito de validação das entradas do usuário.

2.3 Validações

A entrada correta de dados é requisito mínimo para o bom funcionamento do programa. Diante disso as validações são essenciais, por isso as entradas, exceto nome e contato dos doadores, foram protegidas de possíveis erros do usuário. Todas as validações operam de maneira semelhante, com um *loop* usando *while,* onde o utilizador do programa ficará preso até colocar uma entrada válida.

Como exemplo de como as validações foram feitas (Figura 4) será explicado em detalhes a validação da entrada da quantidade de doações. Inicialmente foi determinado que a quantidade de doações não pode ser um dígito, por isso foi usada a função *isnumeric,* pois enquanto a entrada for qualquer coisa diferente de um número a mensagem "Resposta inválida" irá aparecer. A segunda parte da validação está relacionada à quantidade doada, já que não se pode permitir uma doação negativa. Em vista disso a entrada, inicialmente em *string*, foi transformada em um número de ponto flutuante e verificada se é ou não um número negativo. Logo após a mensagem de erro é pedido novamente a entrada do dado.

```
quantidade = input('Quantidade em quilogramas de açúcar: ')
while not (quantidade.isnumeric()) or float(quantidade) < 0:
    print('Resposta inválida.\n')
    quantidade = input('Quantidade em quilogramas de açúcar: ')
```

Figura 4

2.4 Saídas de dados

As saídas de dados são somente a apresentação de todas as variáveis contadoras que acumularam valores ao longo do programa. A única exceção a essa regra é justamente a contagem de quantas cestas básicas poderão ser formadas, uma vez que não foi feito relatório parcial neste programa.

A ideia de como contabilizar as cestas surgiu em uma das sessões do PBL e consiste em usar um *while* com os valores mínimos de cada item para se fazer uma cesta e ir subtraindo a quantidade até não for mais possível entrar no *loop*, lembrando que existe uma variável contadora do total de cestas dentro desse *while*.

Para a contabilidade de cestas com e sem itens extras foi feita uma comparação usando um *if* a fim de descobrir qual é o valor limitante, ou seja, o que vai limitar a produção de cestas com itens extra sendo ele, o número de cestas, ou de itens extra. Na primeira situação o número de cestas formadas é maior ou igual ao número de itens extras, nesse caso o valor limitante é o de itens extras. Na primeira possibilidade o valor de cestas básicas com itens extras será exatamente o valor dos itens extras.

Na segunda situação o valor limitante é o valor de cestas formadas, ou seja, o valor de cestas básicas com itens extras será o número de cestas básicas. Ao final dessa segunda hipótese o número de cestas com itens extras será o número de cestas básicas formadas. Tendo salvo o valor de cestas básicas com itens extras em uma variável chamada cestas_com_extra basta uma subtração para obter o valor de cestas sem itens extra e a quantidade restante de itens extra.

Ao final é apresentado ao usuário o que restou de todos os itens após a formação das cestas básicas. Existem diversas formas de formatar a saída de dados em Python, mas foi escolhida essa por uma questão de costume do programador (Figura 5). A

última entrada de dados é justamente uma pergunta se o usuário deseja ou não ver a lista de nomes e contatos dos doadores, mesmo não sendo obrigatório foi colocado somente para não se perder dados que possam ser importantes para uma possível consulta.

```
# Atualizando para o usuário a quantidade de itens restantes depois que formou as cestas.
print('Quantidade de itens restantes:\n\nAçúcar em Kg: %.2f\nArroz em Kg: %.2f\nCafé em Kg: %.2f\n'
      'Extrato de tomate unidades: %d\nPacote de macarrão unidades: %d\nPacote de bolacha unidades: %d\n'
      'Óleo em litros: %.2f\nFarinha de trigo em Kg: %.2f\nFeijão em Kg: %.2f\nSal: %.2f\n'
      'Itens extras unidades: %d\n' % (quantidade_acucar, quantidade_arroz, quantidade_cafe,
       quantidade_extrato, quantidade_macarrao, quantidade_bolacha, quantidade_oleo, quantidade_farinha_trigo,
       quantidade_feijao, quantidade_sal, quantidade_extra))
```

Figura 5

3. Resultado e discussões

Esse item visa discutir a fundo algumas questões importantes para o entendimento do programa e até mesmo os seus erros e falhas.

3.1 Manual de uso

Para ter acesso ao programa é necessário ter o arquivo do programa, visto que ele não está disponível para ser baixado. Ademais é necessário ter o Python instalado no computador ou qualquer IDE com o Python incorporado. Ao abrir o arquivo é necessário colocá-lo para executar, automaticamente o programa irá iniciar as perguntas necessárias para o cadastro. É fundamental ressaltar que ao fechar o programa todo e qualquer dado colocado será perdido.

3.2 Testes

Os testes foram feitos ao longo da construção do programa para verificar a funcionalidade do mesmo e também se as validações foram feitas de forma correta. Visto que o código foi testado diversas vezes não será possível constar neste relatório todos os testes, mas o principal deles foi justamente a contagem das cestas. Colocando um mesmo valor de entrada para todos os itens foi possível analisar o funcionamento correto da contagem de cestas básicas (Figura 6). Ademais, as validações de dados foram testadas à exaustão para não ocorrer a quebra do código.

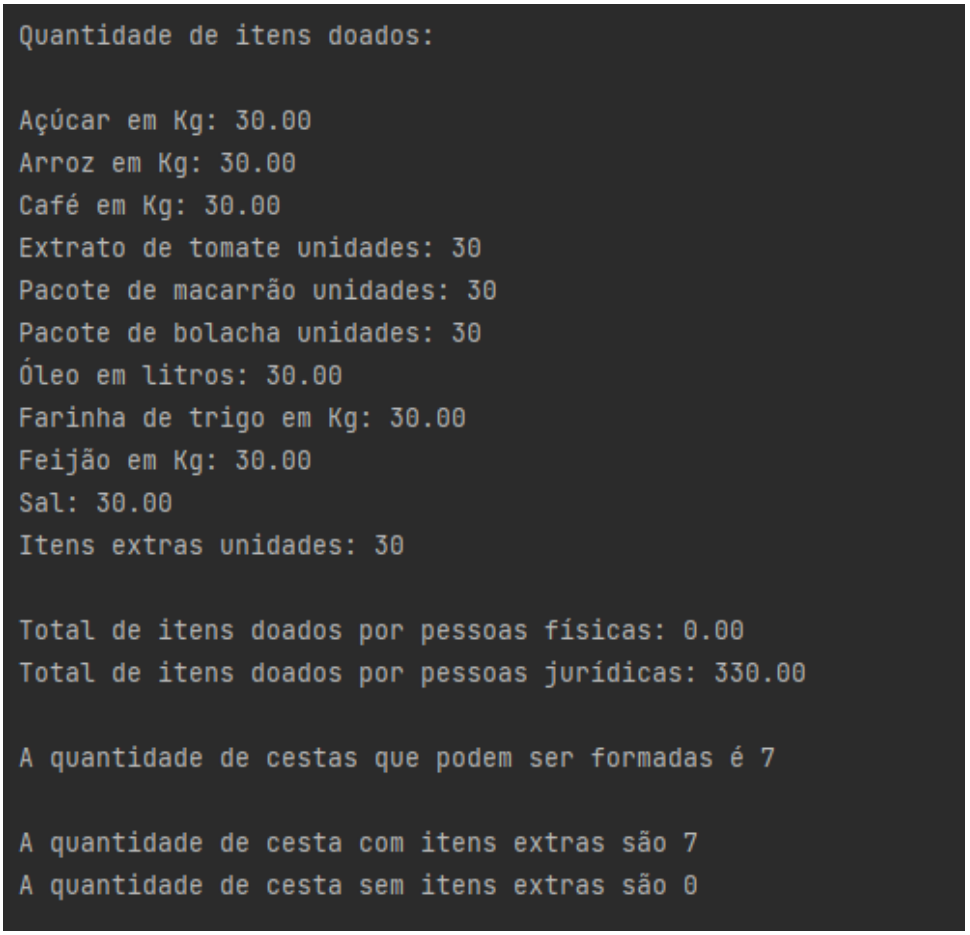


Figura 6

3.3 Erros e melhorias

Um dos defeitos do programa apresentado é justamente a indução de que o usuário somente irá usá-lo para cadastrar doações, não podendo assim abri-lo para consulta. Além do mais, caso se cadastre algum doador e o mesmo não apresentar nenhum alimento para doar o sistema o incluí na lista de doadores. Ainda que apresentar a lista de doadores e contatos ao final do programa não é obrigatório, isso pode ser considerado um defeito.

Por fim, o último problema do programa a ser pontuado não é de mal funcionamento e sim de otimização. A variável quantidade que contabiliza a quantidade de itens doados aparece 11 vezes no código, mas ela aparece acima de cada variável contadora de cada tipo de alimento. Seria possível sintetizar todas as variáveis em uma única linha, porém a mensagem que aparece necessariamente teria que ser genérica (Figura 7). Não foi feito da forma mais otimizada por uma questão puramente estética, a mensagem que aparece reafirma aquilo que o usuário colocou como opção para a doação.

```
elif item == 9:
    quantidade = input('Quantidade em quilogramas de feijão: ')
    while not (quantidade.isnumeric()) or float(quantidade) < 0:
        print('Resposta inválida.\n')
        quantidade = input('Quantidade em quilogramas de feijão: ')
```

Figura 7

4. Conclusão

O objetivo deste projeto é desenvolver um programa capaz de auxiliar o Dispensário Santana a organizar e categorizar toda e qualquer doação de alimento recebida. De resto colocar em prática todos os conhecimentos a respeito de estruturas condicionais e de repetição vistos em sala de aula.

De fato, os dois objetivos principais deste trabalho foram atendidos, uma vez que o programa funciona bem e os alunos puderam colocar em prática todo o conhecimento adquirido até o momento.

Porém o programa poderia sim ser melhorado, a exemplo de salvar os dados em algum arquivo para poder acessar futuramente ou até mesmo apresentar ao final uma única lista de nome e contato dos doadores lado a lado. Esses dois tópicos não foram obrigatórios, mas se fossem feitos iam deixar o programa mais funcional.

5. Referências

Martelli, Alex (2006) “Python in a nutshell”, 2. ed. Beijing; Sebastopol, CA: O'Reilly.

“A história do Dispensário Santana” (2010) http://dispensariosantana.blogspot.com/p/historia.html, Acessado em 23 de março de 2022.

https://www.dispensariosantana.com.br/, Acessado em 23 de março de 2022.

“Dispensário Santana: 36 anos transformando vidas”https://www.feiradesantana.ba.gov.br/servicos.asp?titulo=Dispens%E1rio%20Santana:%2036%20anos%20transformando%20vidas%20&id=10&link=secom/noticias.asp&idn=22316#:~:text=Inaugurado%20em%2029%20de%20maio,presta%20v%C3%A1rios%20servi%C3%A7os%20%C3%A0%20comunidade, Acessado em 23 de março de 2022.

Lima, Guilherme (2021) “Python: A origem do nome” https://www.alura.com.br/artigos/python-origem-do-nome, Acessado em 30 de março de 2022.