Sistema para o Censo Demográfico de 2020

Eduardo Fernandes de Brito Pereia[1]

[1]UEFS – Universidade Estadual de Feira de Santana Av. Transnordestina, s/n, Novo Horizonte Feira de Santana – BA, Brasil – 44036-900

duarduz@gmail.com

Resumo. *Este relatório descreve o problema proposto pelo componente obrigatório, do primeiro semestre, MI Algoritmo I, o qual retrata a solicitação do IBGE para a implementação de um programa que colete as respostas da pesquisa do Censo Demográfico. O simulador, portanto, coleta as respostas e as computa a fim de estabelecer estatísticas que serão apresentadas ao usuário final. Para implementação do simulador, foi necessário desenvolver um programa na linguagem Python, sendo baseado em percorrer arquivos que contém as respostas, onde são utilizados listas e dicionários para processar e armazenar as respostas. Assim, o software desenvolvido simula o resultado do Censo.*

1. Introdução

Segundo o IBGE (2019), o Censo Demográfico "constitui a principal fonte de referência para o conhecimento das condições de vida da população em todos os municípios do País e em seus recortes territoriais internos, tendo como unidade de coleta a pessoa residente, na data de referência". O IBGE decidiu iniciar um projeto piloto para que a coleta destes dados seja feita através de um sistema automatizado. Para tanto, solicitou esse piloto dos estudantes do MI de Algoritmos, do curso de Engenharia de Computação da UEFS.

Este relatório objetiva descrever um sistema desenvolvido na linguagem de programação Python, como solução para o problema anterior. O sistema serve para analisar as respostas dada na pesquisa a fim de criar estatísticas e apresenta-las a um usuário.

Para resolver este problema, algumas questões precisaram ser estudadas: 0) como abrir um arquivo e percorre-lo, 1) como armazenar as informações contidas naquele arquivo a fim de calcular as estatísticas posteriormente, 2) como percorrer uma estrutura de dados a fim de calcular as estatísticas solicitas pelo usuário, 3) quais estatísticas devem ser calculadas e como são calculadas, 4) como mostrar o resultado dessas estatísticas para o usuário.

A solução é organizada da seguinte forma:

1. Os arquivos necessários para resolução do problema são abertos e transformados em dicionário;
2. Verificações são feitas para que só sejam aceitos arquivos validos
3. Esses dicionários são percorridos a fim de calculas as estatísticas

4. Num loop um menu é apresentado ao usuário para que ele escolha qual estatística deseja visualizar.
5. A estatística é apresentada ao usuário;
6. O programa pergunta ao usuário se ele deseja visualizar outra estatística;

3. Desenvolvimento

Nesta seção, procura-se apresentar como as questões anteriores elencadas na introdução foram tratadas na solução do problema.

Como abrir um arquivo e percorre-lo? O censo ou recenseamento demográfico é um estudo estatístico referente a uma população que possibilita o recolhimento de várias informações, tais como o número de homens, mulheres, crianças e idosos, onde e como vivem as pessoas. Essas informações são processadas e cálculos são feitos a partir delas a fim de calcular estatísticas traçando um perfil daquelas pessoas pesquisadas. No software a coleta das respostas é feita a partir de 3 arquivos de texto, 1 contendo as informações dos técnicos que aplicaram a pesquisa, outro com as informações das cidades em que essa pesquisa foi aplicada e o último arquivo contém as respostas dos usuários.

Para abrir um arquivo, o Python possui a função open(). Ela recebe dois parâmetros: o primeiro é o nome do arquivo a ser aberto, e o segundo parâmetro é o modo que queremos trabalhar com esse arquivo - se queremos ler ou escrever. O modo é passado através de uma *string*: "w" para escrita e "r" para leitura.

No software desenvolvido, foi utilizado a função open() com o modo de leitura,

Dessa forma poderemos percorrer o arquivo e o armazenar em alguma estrutura de dados para fazer os cálculos posteriormente.

Como armazenar as informações contidas naquele arquivo a fim de calcular as estatísticas posteriormente? Com o arquivo aberto, precisamos percorre-lo para armazena-lo em uma estrutura de dados que facilite os cálculos posteriores, dessa maneira utilizamos a função readline() que lê linha por linha do arquivo e o armazena em uma variável do tipo list, desta forma podemos utilizar essa função em um for do tamanho de linhas do arquivo para que todas as linhas do arquivo sejam lidas e armazenadas, porém, a melhor estrutura de dados para se trabalhar nesse software é a estrutura dicionário que são estruturas de dados que implementam mapeamentos, um mapeamento é uma coleção de associações entre pares de valores, um dicionário é disposto como {Chave:Valor}, facilitando assim busca de valores.

Desta maneira, nesse software funciona transformando a variável tipo list da função readline() em um variável tipo dict, que é o tipo da estrutura dicionário.

Quando estamos trabalhando com arquivos, devemos nos preocupar em fechá-lo. Para fechá-lo usamos a função close(), desta maneira quando o software percorre todo o arquivo e transforma-o em um dicionário, a função close() é chamada e o arquivo é fechado.

Como percorrer uma estrutura de dados a fim de calcular as estatísticas solicitas pelo usuário? Dicionários são estruturas poderosas e muito utilizadas já que

podemos acessar seus elementos através de chaves e não de sua posição. Em outras linguagens este tipo é conhecido como "matrizes associativas".

Qualquer chave de um dicionário é associada (ou mapeada) a um valor. Os valores podem ser qualquer tipo de dado do Python. Portanto, os dicionários são pares de chave-valor não ordenados.

Tendo isso em vista, o software, num laço de repetição for com o método get(), obtém o conteúdo de todas as chaves facilitando os cálculos das estatísticas.

Quais estatísticas devem ser calculadas e como são calculadas? No problema são solicitadas 7 estatísticas 1) Números de domicílios utilizados para a coleta; 2) Número de domicílios particulares que já estão pagos, quantos ainda estão pagando e alugados; 3) Quantos domicílios por cidade possuem banheiro e quantos não possuem; 4) A forma mais comum de abastecimento de água por cidade; 5) O percentual de domicílios por cidade que ainda não possuem energia elétrica; 6) O percentual de moradores que participaram da entrevista por cor ou raça. 7) A região com maior número de municípios pesquisados.

1. Função com um laço de repetição com um acumulador que conta quantas vezes o método readline() foi chamado.
2. Função que busca as respostas 2.01 do arquivo “exemploPesquisa” e acumula numa lista o número de domicílios pagos, que ainda estão pagando e quantos estão alugados.
3. Função que busca as resposta 2.02 do arquivo “exemploPesquisa” e acumula num dicionário onde as chaves são as cidades e os valores são o número de domicílios que não tem banheiro e o número de domicílios com banheiro encontrados naquela cidade.
4. Função que busca as respostas 2.05 do arquivo “exemploPesquisa” e acumula num dicionário onde as chaves são as cidades e os valores são a forma de abastecimento daquele domicilio que situa-se na cidade chave.
5. Função que busca as respostas 2.07 do arquivo “exemploPesquisa” e acumula num dicionário onde as chaves são as cidades e o valores são o número de domicílios sem energia, o número de domicílios que possuem energia através de distribuidora e o número de domicílios que possuem energia através de outras formas.
6. Função que busca as respostas 4.03 do arquivo “exemploPesquisa” e acumula num dicionário onde as chaves são as raças e os valores são a quantidade de pessoas daquela raça na pesquisa.
7. Função que busca os estados em que a pesquisa foi aplicada e os separa em um dicionário onde as chaves são as regiões do país e os valores são o número de cidades daquela região.

Como mostrar o resultado dessas estatísticas para o usuário? Na linguagem, de tipagem dinâmica, Python, comandos de saída, como print, permitem a exposição de mensagens na tela, já comandos de entrada, como input, permitem a exposição de alguma mensagem, de forma optativa, e recebem dados, os quais são utilizados com variáveis a fim de armazenar o valor digitado [Devmedia 2017].

Dessa forma, o software disponibiliza, num laço de repetição um menu com opções para que o usuário escolha se deseja visualizar alguma estatística ou deseja sair do programa.

2.1. Organização do código fonte

O código é dividido em várias funções com finalidades especificas numa tentativa de manter o código com alta coesão e baixo acoplamento.

3. Conclusões

Este relatório apresentou uma solução de software para similar o Censo Demográfico. O simulador foi implementado em um programa na linguagem Python, sendo os dados armazenados em memória. A solução resolve o problema proposto, fazendo uso de várias construções de comandos, como os de entrada e saída, estruturas de repetição e estruturas de dados.

Vários subproblemas foram explicitados neste relatório e suas soluções foram mostradas na seção de desenvolvimento.

O software resolve o problema proposto, atendendo as especificações solicitadas pelo cliente. Entretanto, serve apenas para o Censo e apresenta uma limitação grande por ter de guardar os dados em memória, os erros são tratados apenas linha por linha. Versões futuras podem adicionar mais flexibilidade pelo uso de arquivos para os questionários e para guardar as respostas.

Os pontos fortes deste relatório são a organização detalhada do problema em subproblemas e de suas soluções. Por outro lado, pode haver problemas de redação que reduzam a clareza e a concisão.

Referencias

DevMedia (2017). Python: Trabalhando com Variáveis. https://www.devmedia.com.br/python-trabalhando-com-variaveis/38644.Acessado em 06 Agosto. 2019.