A enquete mais flexível deste lado do Mississippi.

Daniel Fernandes Campos

Engenharia de computação – Universidade Estadual de Feira de Santana (UEFS)

CEP: 44036-900 – Feira de Santana – BA – Brasil

dfc152@gmail.com

Resumo. *Com a necessidade de realizar diversas foi solicitado que fosse um algoritmo capaz de receber alguns dados do pesquisador de arquivos de texto e fossem utilizados para fazer uma pesquisa, que pudesse ser continuada posteriormente. Para criação deste código foi necessário o aprendizado de alguns temas e discussões em grupo sobre como resolver o problema e subproblemas para então escrever o script. Código este que realiza a pesquisa com base em arquivos de texto e apresenta escore (média), moda e frequência absoluta para as perguntas respondidas.*

1. Introdução

Com a necessidade fazer diversas pesquisas o Núcleo de Estudos da Democracia da UEFS (NED-UEFS), desejando ampliar a parceria com o curso de Engenharia da computação solicitou que fosse criado um algoritmo, que será utilizado para realização de pesquisas. Este relatório tem como objetivo relatar está criação, bem como apontar falhas e funcionalidades deste código.

Para criação deste *script* foi necessário utilizar a manipulação de arquivos de texto, desde criar e modificar a como utilizar como entradas pois foi pedido que as pesquisas pudessem ser mais flexíveis e para modificação exigir uma menor noção sobre programação do pesquisador. Houve a necessidade de entender o funcionamento de vetores de variáveis do tipo char, bem como entender suas peculiaridades, para fazer a manipulação dos dados (questões e escalas) que o pesquisador escreveria no .txt, e também o funcionamento de ponteiros, para ser feito um código modularizado, ou seja, mais agradável a vista e mais fácil de manuseamento.

Durante a criação deste algoritmo foi necessário a realização de pesquisas sobre os temas acima citados, discussões sobre como realizar partes do código com colegas e algumas horas de codificação.

2. Desenvolvimento

O primeiro passo é entender o problema por isso foi feito um fluxograma para dar um norte para a resolução deste problema (Figura 1), assim podemos subdividir cada parte deste fluxograma em um problema menor. Problemas estes que poderão ser subdivididos novamente assim tornando um problema mais simples, ideia esta que foi dada durante uma das seções de discussão do problema, assim sendo o código foi dividido em diversos procedimentos para realização de cada subproblema.

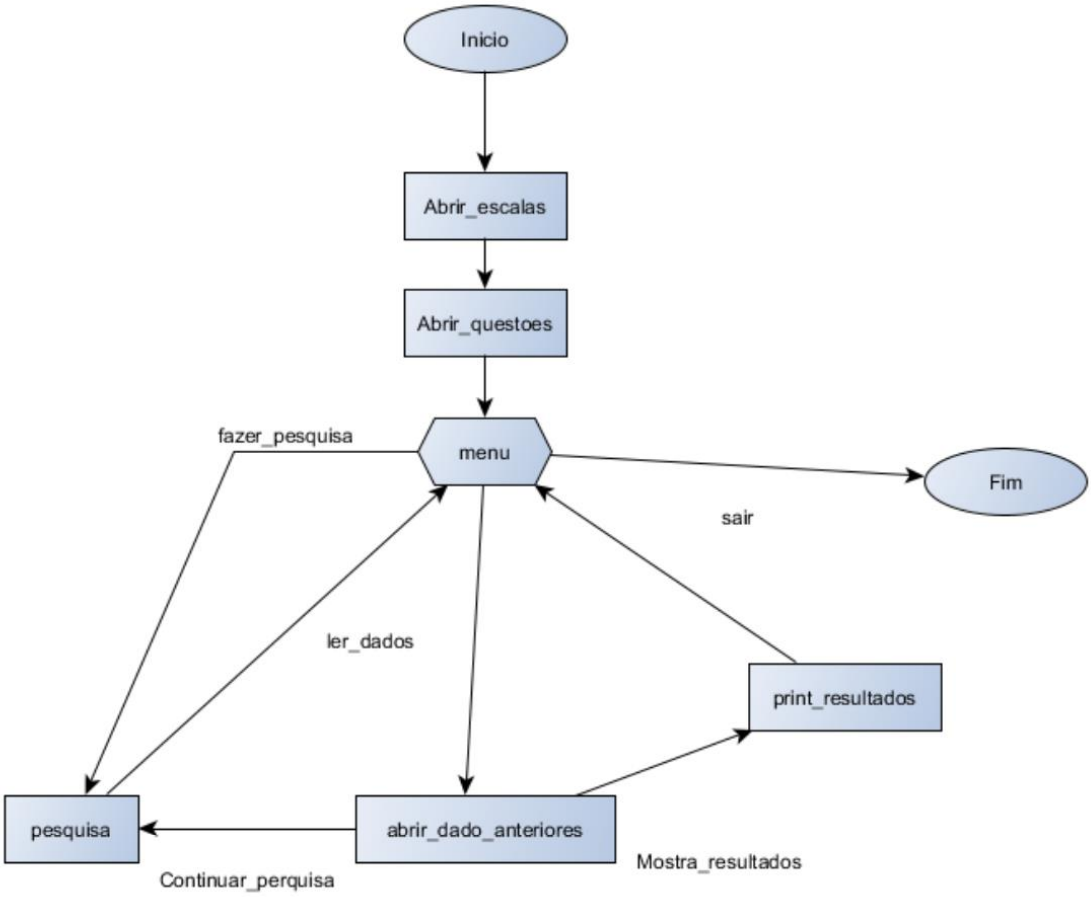

Figura 1. Fluxograma alto nível sobre o problema.

Partindo disso foi necessário aprender sobre como manipular arquivos .txt em c. Para fazer a leitura dos arquivos foram usadas funções (figura 2), nas quais o arquivo era aberto e os dados eram alocados em umas struct. Uma struct é algo bastante similar a um vetor só que com caracteres não homogêneos, ou seja, é um vetor no qual é escolhido o que cada elemento deste vetor representa (figura 3).

```c
int abrir_escala( char nome_arquivo[101] , int quantidade, int  *contador_quantidade_escalas, struct geral *escala){
    setlocale(LC_ALL, "Portuguese");
    int i =0;
    FILE *file;
    file = fopen (nome_arquivo, "r") ;
    if( file == NULL){
        printf("arquivo não aberto! algo inesperado aconteceu!!!");
        return 0;
    }
    char str1[350], str2[350];
    while(feof(file) == 0 && *contador_quantidade_escalas < quantidade){
        fscanf (file, "%i", &escala[i].numero_da_struct);
        fscanf (file, "%i", &escala[i].codigo_usado);
        fgetc(file);
        fgets(escala[i].texto_total, 350, file);
        *contador_quantidade_escalas +=1;
        i++;
    //  printf("escala %i pega\n", *contador_quantidade_escalas); <---- para dbug caso tenha problema com escala serve para ver quantas escalas pegas
    }
    fclose(file);
    return 1;
}
```

Figura 2. Imagem da função criada para abrir escalas..

```c
struct geral { // struct base para ser usada para escala e questao
    int numero_da_struct; // conta o numero da qeustao ou escala
    int codigo_usado; // codigo da escala ou codigo chamado da questao
    char texto_total[350]; // texto total que sera separado se escala
    char escala_resposta[10][30];
};
```

Figura 3. Exemplo de struct do algoritmo.

Como foi mostrado na figura 2 está sendo usado matrizes 2D, que são “vetores de vetores”, ou seja, um agrupamentos de um tipo de variável fundamental em uma dada ordem, “escala_resposta” é um exemplo de matriz 2D.

Devido as escalas usadas serem colocadas todas em uma linha do bloco de notas foi necessário fazer uma função que separar cada uma das escalas em um elemento da “escala_resposta” presente na struct, para que quando for pintar questão e sua escala correspondente fosse mais fácil. A utilização de *struct* foi uma escolha pessoal, sendo também possível utilização de uma matriz 2D e 3D.

Após esta etapa comecei a fazer a pesquisa, uma função que realizaria todo o script, chamando outras se necessário. A função pesquisa começar chamando outra função em que é feito uma “triagem” do candidato. Esta triagem pega o curso, sexo e idade do candidato, foi solicitado que no arquivo estivessem escritos o curso e o sexo do candidato por extenso, portanto para isso foi disponibilizado um modelo de como a resposta deve ser escrita para o usuário digita-la. Em seguida é printando a pergunta e então chamado outra função que irá procurar a escala desta questão e era printar elemento por elemento da escala seguido pelo número correspondente a estas respostas.

Com todas as perguntas respondidas os dados do candidato são salvos no arquivo e é printado o escore das respostas do indivíduo, seguido de uma pergunta se deseja fazer a pesquisa com outro candidato, caso sim volta ao início da função, caso contrário segue para o menu.

A opção de continuar pesquisa do menu irá abrir os resultados já presentes no arquivo alocar em um vetor de struct de entrevistados e seguir para pesquisa. A função de abrir resultados também é chamada para quando a opção de ver resultados é selecionada. Para a opção ver resultado foi feito 2 matrizes, uma para guardar as respostas para calcular os escores e outra que era um contador das respostas (para moda) com informação de todos os entrevistados.

Um dos problemas enfrentados na criação deste código foi para fazer a filtragem dos candidatos que seriam considerados para calcular os escores, modas e médias finais. Inicialmente foi pensado fazer a filtragem utilizando int (códigos numéricos) mas foi logo descartada por a dificuldade de integrar o número a uma string no arquivo, apesar de impedir erros de digitação, mas consumiria bastante linhas de if para isso e para acrescentar um curso novo curso seria muito trabalhoso.

Portanto foi utilizado uma *string* e pedido que o usuário digite tanto na triagem do entrevistado quanto na da geração de dados da pesquisa. Porém com a utilização desse sistema em algum lugar do processo (não identificado, provavelmente a do arquivo) alguma *string* continha um ‘\n’ adicional e para isso foi feito tanto na leitura nas triagens quanto na leitura do arquivo um loop para procurar se existe e caso exista remover. O processo anteriormente citado não foi feito através de função por algum bug que estava acontecendo quando o caractere da *string* passada por ponteiro era checado.

3. Conclusão.

Para a produção deste código o requisito não alcançado foi o cálculo do escore feito corretamente, os outros foram alcançados (calcular moda, fazer a pesquisa e salvar em .txt, continuar pesquisas, print de frequências absolutas).

Algumas das melhorias que poderiam ser feitas no código seriam a implementação de uma interface mais agradável ao usuário, a reformulação do cálculo do escore, solicita do entrevistador em qual arquivo ele quer salvar aquela entrevista e os cursos presentes na pesquisa fossem colocados em um .txt para que o pesquisador pudesse escolher quais ele quer que estejam e quais ele não quer.

Referências

Sarroglia, Marcio Pinho Ober. **Programação em c/c++**: uso de arquivo de texto. Available on:<https://www.inf.pucrs.br/~pinho/LaproI/Arquivos/Arquivos.htm#Gravacao>. Acesso em: 12/11/2018

cplusplus. **Manipulação de arquivos em C**. Available on: <http://wiki.icmc.usp.br/images/8/82/Manipulacao_arquivos.pdf>. Acesso em: 12/11/2018.

Feofiloff, Paulo. (2018) **Registros e structs**. Available on: <https://www.ime.usp.br/~pf/algoritmos/aulas/stru.html>. Acesso em: 12/11/2018.