UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ARTES, CIÊNCIAS E HUMANIDADES

**Exercício de Programação: Concatenação de Vetores.**

Trabalho apresentado na disciplina ACH2055 - Arquitetura de Computadores

Autores: Renan Rogério Boni 8921238

Arthur Ferreira 9277051

Pedro Beraldo 6777363

Thiago S. de Sá M. Bastos 9004459

São Paulo

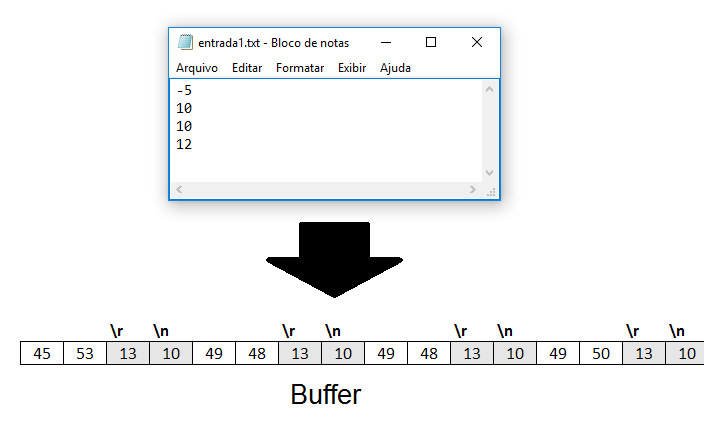
22 de Outubro de 2016

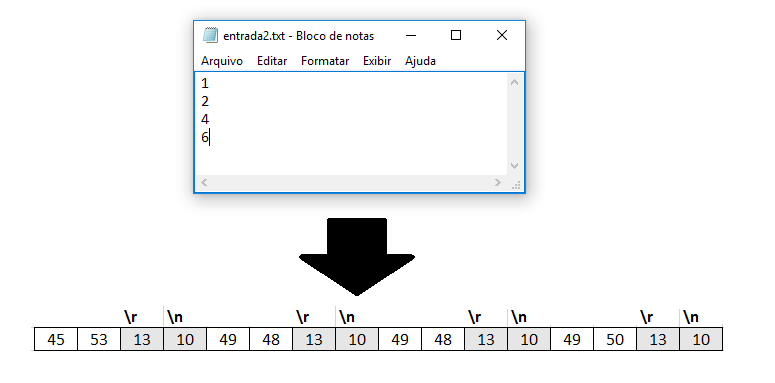
# Leitura

Na leitura dos arquivos, o procedimento *open\_file* será responsável por ler os dois arquivos e armazenar o conteúdo de ambos em um buffer separado. (A leitura do arquivo é feito byte a byte).

A primeira etapa da leitura envolve armazenar o conteúdo do arquivo em um buffer com um tamanho pré-definido e em seguida percorrer cada arquivo para obter seu tamanho (o tamanho se refere a quantidade de elementos no arquivo, no caso de uma lista contendo números, basta contar o número de espaços entre eles e a quebra de linha, que indica o final do arquivo).

Observe que por mais que o arquivo contenha números, eles são interpretados como caracteres (de acordo com a tabela ASCII), logo a conversão é semelhante à figura abaixo:

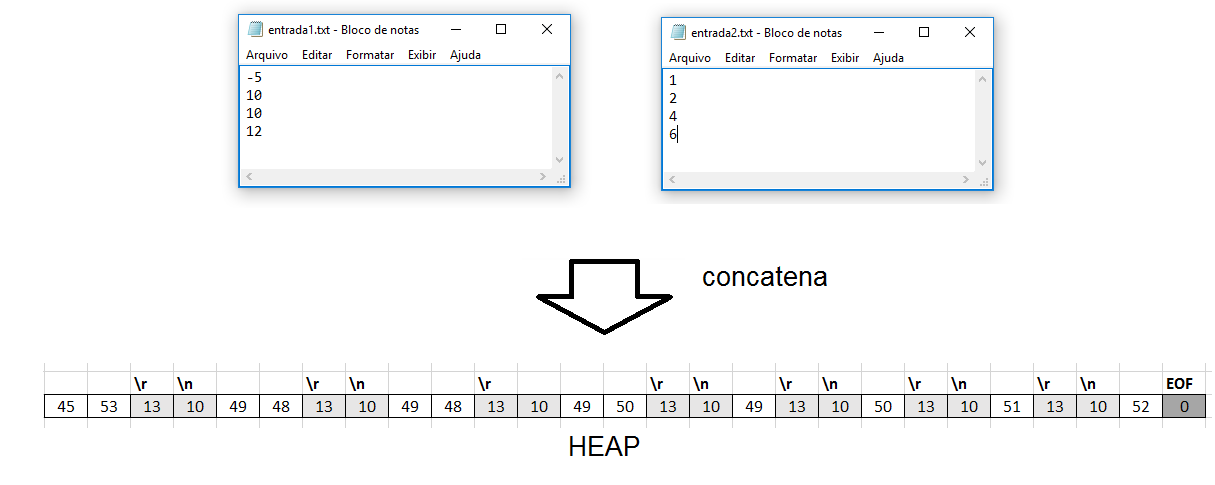


O mesmo procedimento é feito para o segundo arquivo, que será salvo em um buffer distinto.

# Concatenação

O procedimento de concatenação recebe por parâmetro o buffer de cada vetor e os seus respectivos tamanhos e então, concatena as duas listas utilizando agora a memória heap e então guarda o endereço utilizado para salvar na memória.

Assim, para a concatenação dos dois arquivos lidos, será semelhante ao exemplo da figura abaixo:



# Conversão

Agora que já temos a lista concatenada, é necessário converter o inteiro em seu valor correspondente seguindo a tabela ASCII. Então, este procedimento é responsável por varrer a lista e converter cada posição em seu respectivo valor (Este procedimento é semelhante a função atoi da linguagem C).

Assim teremos o seguinte resultado:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -5 | 10 | 10 | 12 | 1 | 2 | 4 | 6 |

# Ordenação

Como unimos duas listas e decidimos por concatenar, por mais que elas já estivessem em ordem, ao colocar uma ao final da outra torna-se necessário a ordenação de toda a lista.

Então o procedimento ordenação recebe como parâmetro a posição inicial da lista, a última posição e o endereço em que a mesma está.

O algoritmo de ordenação utilizado é o Quick Sort, no qual adota a estratégia de divisão e conquista, abaixo temos o algoritmo para sua execução (retirado do *Wikipedia*):

1. Escolha um elemento da lista, denominado pivô;
2. Rearranje a lista de forma que todos os elementos anteriores ao pivô sejam menores que ele, e todos os elementos posteriores ao pivô sejam maiores que ele. Ao fim do processo o pivô estará em sua posição final e haverá duas sublistas não ordenadas. Essa operação é denominada partição.
3. Recursivamente ordene a sublista dos elementos menores e a sublista dos elementos maiores.

Sua complexidade no melhor caso e no caso médio é nlogn e no pior caso, n².

Após a execução do procedimento, teremos o seguinte estado em nosso vetor:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -5 | 1 | 2 | 4 | 6 | 10 | 10 | 12 |

# Remoção dos repetidos

Após a ordenação da lista, é possível obtermos a ocorrência de elementos repetidos no vetor final, o que não faria sentido para nossos objetivos, então é necessário este procedimento que ficará responsável por eliminar os elementos repetidos, que basicamente funciona da seguinte maneira:

Cada elemento é comparado com os seus sucessores e em caso de o seu sucessor for igual ao elemento comparado, então toda a lista é movida uma casa para esquerda, “engolindo” o elemento repetido. Logo, para nosso exemplo, teremos:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| -5 | 1 | 2 | 4 | 6 | 10 | 12 |

# Escrita

Para a escrita no arquivo, o procedimento *save\_file* possui rotinas que convertem cada digito para seu respectivo valor na tabela ASCII e então adiciona em um buffer temporário, em seguida, uma quebra de linha é adicionado para gerar a formatação pedida.

# Considerações finais

É válido ressaltar que o problema foi resolvido usando o sistema operacional Windows 10, no qual apresenta algumas diferenças na representação de caracteres em relação a outros sistemas, como por exemplo, a representação de quebra de linha, no Windows se dá por “\r\n”, no Linux, somente “\n” e então no iOS por “\r”.