ACH2024 ALGORITMOS E ESTRUTURAS DE DADOS II

Semestre 2015-1 - Exercício prático - Processamento cossequencial

Responsável - Ivandré Paraboni (ivandre@usp.br)
Estagiário PAE - Adilson Khouri (adilson.khouri.usp@gmail.com)

1. O objetivo do trabalho é implementar de forma correta e completa uma função de a ordenação de um arquivo extenso do tipo texto usando apenas um vetor de tamanho limitado que simula a memória disponível.
2. Verifique no Tidia o modelo de código com as definições (*typedef*) que você deve utilizar.
3. O arquivo de entrada é uma série de inteiros desordenados e possivelmente repetidos, representados em formato texto e separados por espaços em branco. Verifique o arquivo de exemplo (Testes.zip) disponível no Tidia, mas lembre-se de que é apenas um exemplo.
4. A função recebe como entrada um parâmetro *char *nome* que indica o nome de um arquivo a ser ordenado. A saída da função é a criação de um arquivo de nome *saida.txt* contendo os mesmos dados de entrada, porém **em ordem crescente e sem repetição**.
5. A assinatura da função é a seguinte:

   void ordenar(char *nomearq)

6. O objetivo deste exercício é efetuar a ordenação de arquivos utilizando como estrutura de dados apenas o vetor int M[100] declarado no código exemplo, e que representa o máximo de memória disponível para os dados. Assim, a ordenação deve usar corridas de no máximo 100 inteiros cada.
7. Não há restrição quanto às técnicas de ordenação interna (Quicksort, Bubblesort etc.) e externa (monofase, multifase etc.) utilizadas desde que a função não use mais memória do que a capacidade do vetor M. Além do vetor M, apenas variáveis simples são permitidas.
8. Lembre-se: não há memória disponível além de M[100]. Seu código não pode conter nenhum tipo de lista auxiliar (vetor, lista ligada, árvore etc.), nem fazer alocação dinâmica de nós.
9. Não use variáveis globais. A função implementada deve definir localmente todas as variáveis e estruturas auxiliares, ou chamar funções auxiliares que o façam também em um escopo local.
10. Não exiba nenhuma mensagem na tela, nem solicite que o usuário pressione nenhuma tecla etc. Apenas implemente a função solicitada.
11. A função main() serve apenas para seus testes particulares, e não precisa ser entregue. Caso você prefira mantê-la no corpo do programa, pede-se apenas que main() seja a última função do programa, ou seja, que não haja nenhum código abaixo dela.
12. Arquivos temporários podem ser criados em qualquer quantidade, mas sempre com a extensão .tmp. Não crie arquivos com outro tipo de extensão, ou em outras pastas.
13. Não apague os eventuais arquivos temporários que forem criados. Este procedimento é dependente de sistema operacional, e se você implementá-lo há o risco do seu programa não passar nos testes em outro ambiente.
14. Todos os arquivos temporários e o resultado final devem ser criados na pasta atual onde o programa está sendo executado. Não crie subpastas.
15. O EP pode ser desenvolvido individualmente ou em duplas.
16. Não tente emprestar sua implementação para outros colegas, nem copiar deles, pois isso invalida o trabalho de todos os envolvidos.
17. 0 programa deve ser compilável no Dev-C++ ou Codeblocs sob Windows 7 ou superior.
18. Programadores JAVA, cuidado: não existe inicialização automática de variáveis em C.

O que/como entregar:

- A entrega será via upload no sistema Tidia por apenas um integrante de cada dupla.
- Entregue apenas o código da função principal e das funções auxiliares que ela invoca como anexo de nome XXXXXX_YYYYYY, onde XXXXXX e YYYYYY são os números USP dos desenvolvedores.
- A extensão do arquivo pode ser .c ou .cpp - favor **não compactar**.
- Preencha as funções nroUSP1 e nroUSP2 do código disponível no Tidia para que você seja identificado. Se o EP for individual, mantenha o valor do segundo nro. como zeros.
- Na primeira linha do código, escreva um comentário "//" com os nomes dos integrantes.

Prazo:

O EP deve ser enviado até **08/06/2015 às 08h00 da manhã**. Não serão aceitos EPs entregues depois do prazo, independentemente do motivo. Entregas no último dia são assim por conta e risco do aluno, e nenhum tipo de imprevisto de última hora (e.g., problemas de saúde, indisponibilidade de rede etc.) pode ser usado como justificativa para o atraso. O EP é uma atividade para ser desenvolvida ao longo de várias semanas, não no último minuto do último dia.

É responsabilidade do aluno que fez o upload do arquivo verificar se o mesmo foi corretamente enviado pelo sistema Tidia.

Critérios de avaliação:

A função será testada com uma série de chamadas consecutivas, com diversos arquivos do tipo fornecido como exemplo. Um teste é considerado correto se o arquivo de saída for exatamente como o esperado, ou incorreto em caso contrário. Erros de alocação de memória ou compilação invalidam o teste, assim como a ausência de funções auxiliares necessárias para a execução do programa.

Este EP deve ser desenvolvido obrigatoriamente por *todos* os alunos de AED2, e sua nota **não é passível de substituição**.