GBC024 - Algoritmos e Estruturas de Dados 1
Trabalho Final de Implementação

1. Objetivo

Construir um sistema computacional que aplique os conhecimentos adquiridos sobre alocação dinâmica de memória e estruturas de dados.

2. Temas

Segue uma lista com sugestões de temas para os sistemas. Os alunos podem propor requisitos e funcionalidades adicionais às especificações apresentadas:

a) **Manipulador de polinômios:** O programa deve ser capaz de manipular polinômios do tipo $P(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x^1 + a_0$. O programa deve possibilitar ao usuário armazenar diversos polinômios criados, atribuindo um identificador a cada um, de forma que ele possa ser selecionado pelo usuário. As seguintes funcionalidades deverão ser disponibilizadas:

1. Inicializar um polinômio da seguinte forma: $P(x) = 0x^0$;
2. Inserir um novo termo $a_k x^k$ no polinômio: caso já exista o termo $a_k x^k$ no polinômio, o programa deve dar ao usuário a opção de i) inserir o termo na posição correspondente, incrementando os indices dos demais termos adequadamente, ii) substituir o termo existente, ou iii) adicionar o valor inserido ao valor já existente no termo ak;
3. Exibir o polinômio $P(x)$, usando a formatação apresentada no início deste enunciado;
4. Eliminar o termo associado à k-ésima potência, fazendo os ajustes necessários para o polinômio continuar válido;
5. Reinicializar um polinômio para a forma: $P(x) = 0x^0$;
6. Salvar o polinômio criado (fica facultativo ao grupo utilizar arquivos ou não para armazenar os polinômios já cadastrados);
7. Somar dois polinômios escolhidos pelo usuário dentre os cadastrados. A soma consiste em somar os termos com expoentes iguais, e copiar de modo intercalado aqueles que não tem correspondente no outro. O novo polinômio poderá ser salvo no programa como os demais;
8. Calcular o valor de $P(x)$ para um valor de x solicitado. Nesse caso, o usuário poderá escolher um ou múltiplos polinômios para o cálculo de $P(x)$. No caso do usuário escolher múltiplos polinômios, o sistema deve mostrar na tela os valores calculados para cada polinômio, um por linha, e mostrar abaixo dos resultados qual polinômio retornou o menor e o maior valor.

b) **Calculadora de números inteiros grandes**: O programa deverá implementar uma calculadora de números inteiros de tamanho arbitrário, sendo capaz de operar com números maiores do que o valor máximo suportado pelo tipos de dados `int` e `long`. Deverá ser criada uma representação para tais números que explore os benefícios oferecidos pelas estruturas de dados vistas em sala. As seguintes operações deverão ser disponibilizadas: + (adição), - (subtração), * (multiplicação), / (divisão inteira) e % (resto da divisão inteira). O programa deve oferecer as seguintes opções ao usuário:

1. Informar os operandos e armazená-los para a operação;
2. Exibir os operandos armazenados;
3. Exibir o menor/maior operando;
4. Realizar o cálculo desejado. Note que, como os operandos já estão armazenados no programa, múltiplas operações (uma de cada vez) poderão ser realizadas com esses operandos. O resultado deverá ser mostrado na tela da seguinte forma: OPERANDO1\n OPERAÇÃO \n OPERANDO2 \n = \n TOTAL.

c) **Gerenciador de pedidos em restaurantes** (ex: *iFood*): O programa deve realizar o cadastro dos restaurantes participantes, bem como dos pratos oferecidos por cada um deles. Cada restaurante possui uma categoria, referente ao tipo de culinária que oferece. Os clientes também são cadastrados previamente, e podem acessar o programa (mediante usuário/senha) para fazer seus pedidos. Os usuários podem pesquisar por tipo de culinária, e o sistema mostra todos os restaurantes cadastrados como resultado. Ao selecionar um restaurante, o usuário seleciona o prato e a quantidade, e o sistema mostra o valor final para pagamento. O usuário deverá entrar no sistema e informar que recebeu o pedido, alterando o status do mesmo. É desejável que o grupo implemente outras funcionalidades no programa, além das citadas, de forma a simular mais fielmente um sistema comercial dessa natureza.

d) **Analisador/Interpretador de fórmulas da Lógica Proposicional:** O programa deverá permitir a criação, validação e interpretação de fórmulas da Lógica Proposicional de acordo com as regras estabelecidas, que podem ser encontradas em [1]. O programa deve oferecer as seguintes funcionalidades:

1. Inicializar uma expressão vazia;
2. Inserir símbolos em uma expressão inicializada (um a um). Nesse caso, o grupo poderá definir um caractere que representa o fim da inserção de símbolos;
3. Validar uma fórmula, de acordo com as regras da Lógica Proposicional;
4. Inserir uma subfórmula já cadastrada em outra fórmula cadastrada. Nesse caso, apenas expressões já validadas como fórmulas poderão ser inseridas como subfórmulas;
5. Cadastrar uma fórmula construída (fica facultativo ao grupo utilizar arquivos ou não para armazenar as fórmulas já cadastradas);
6. Mostrar todas as expressões cadastradas, e sinalizar aquelas que não estão de acordo com as regras da Lógica Proposicional (ou seja, não são fórmulas);
7. Interpretar o resultado da fórmula. Nesse caso, o usuário deve informar o valor verdade de cada um dos símbolos proposicionais que constitui a fórmula, e o programa irá exibir na tela o resultado (**TRUE** ou **FALSE**). Nesse caso, apenas expressões validadas como fórmulas poderão ser interpretadas.

[1] SOUZA, J. N., Lógica para Ciência da Computação e Áreas Afins, versão online: http://www.portal.facom.ufu.br/central-de-conteudos/links/livro-logica-para-ciencia-da-computacao-e-areas-afins

e) **Os alunos poderão sugerir um outro tema diferente dos propostos, que passará por uma avaliação de viabilidade pelo professor.**

3. Regras do trabalho

a. Os alunos deverão fazer o trabalho em grupos de 4 pessoas. Não serão aceitos grupos com menos de 4 pessoas, a não ser que o total de alunos não permita a divisão exata. As listas com os nomes dos integrantes dos grupos, juntamente com o tema escolhido, deverão ser enviados via plataforma Microsoft Teams, na data especificada na **Seção 6**. Após essa data, não será permitido o envio de listas com os nomes dos integrantes, e os alunos que não estiverem em nenhum grupo ficarão com zero na nota final, sem chance para entrega do mesmo;
b. O software deverá ser construído na linguagem C, e deverá utilizar os conceitos de alocação dinâmica de memória e estruturas de dados aprendidos em sala de aula. **Não basta que o software funcione, ele deve fazer uma BOA UTILIZAÇÃO dos conceitos supracitados.**

4. Entrega

a. A entrega dos trabalhos será feita via plataforma Microsoft Teams, na data especificada na **Seção 6**;
b. **O professor não se responsabiliza por falhas na submissão, falhas no MICROSOFT TEAMS, e QUALQUER OUTRO PROBLEMA RELACIONADO. É de responsabilidade do aluno não deixar**

a entrega para última hora, e com isso se prevenir de tais problemas. Não serão aceitas, posteriormente, nenhuma justificativa, e qualquer e-mail de tentativa de justificativa por parte dos alunos será descartado.

c. O sistema pronto deverá ser transmitido em um arquivo compactado (.zip) contendo TRÊS PASTAS:
   i. PASTA 01: códigos-fonte que compõem o software, e/ou estrutura completa gerada pela IDE utilizada;
   ii. PASTA 02: código-fonte compilado (arquivo .exe), de forma que seja possível executar o software sem necessidade de compilação. Este arquivo deve estar SEPARADO da pasta citada no item i;
   iii. PASTA 03: Documentação, contendo:
      1. Arquivo texto contendo os nomes dos integrantes do grupo;
      2. Qualquer biblioteca ou pacote utilizado (CASO SEJA UTILIZADO), **com documentação de como instalá-la, configurá-la e utilizá-la**. Além disso, ao utilizar uma biblioteca, o grupo deverá explicar quais elementos dela são utilizados no sistema, e como são utilizados para o funcionamento do mesmo;

5. Avaliação

a. O trabalho vale 40 pontos, e será avaliado da seguinte maneira:
   - **NOTA-SOFTWARE** → 40,0
     i. Boa utilização dos conceitos aprendidos em sala de aula → 20,0;
     ii. Funcionamento do sistema e corretude das funcionalidades presentes → 20,0 (**caso o programa não execute, a nota referente a esse critério é automaticamente zerada**).

   - **NOTA-APRESENTAÇÃO** → [0 - 1]
     i. Os grupos deverão fazer uma apresentação a respeito do software entregue, mostrando seu funcionamento e respondendo a perguntas sobre sua estrutura;
     ii. A apresentação será feita em duas fases. Na primeira, os alunos mostrarão o sistema funcionando, sem mostrar nenhuma linha de código. Na segunda, o professor questionará cada integrante do grupo separadamente sobre o código. Caso a apresentação seja feita por video conferência, as câmeras deverão permanecer ligadas;
     iii. **Os grupos que não entregarem o software NÃO TERÃO DIREITO À APRESENTAÇÃO;**
     iv. Apenas os integrantes cujos nomes estiverem no arquivo entregue junto ao software terão direito à apresentação, e serão avaliados individualmente;
     v. **Alunos que não apresentarem o trabalho ficarão com nota total do trabalho igual a 0, e a entrega do software não será considerada nesse caso**.

NOTA FINAL = NOTA-SOFTWARE x NOTA-APRESENTAÇÃO

6. Datas Importantes

a. Entrega da lista de integrantes e tema do trabalho: **31/03/2023, ATÉ AS 23:59**
b. Entrega do trabalho: **12/06/2023, ATÉ AS 12:00**;
c. Apresentação: **12/06/2023, 14/06/2023 e 19/06/2023**.