Trabalho Final da Disciplina de Linguagens de Programação – 2020.1 – Modalidade Remota.

Professor João Marcelo Uchôa de Alencar.

Campus da UFC em Quixadá.

Descrição

Este trabalho consiste em diversas questões de programação do livro texto da disciplina. A realização é **individual.** Pedir ajuda aos colegas, observar a solução deles, fazer em equipe etc., é **cola**. Cada questão vale 1,0 ponto.

Data de Entrega: 31/10/2020

Instruções:

Preparação do Repositório.

Crie um repositório **privado** no GitHub. Repositórios públicos levaram nota **zero** automaticamente. Se seu nome for *José Carlos Silva*, o repositório deve ter o nome *josecarloslip20201*. Primeiro e segundo nome, sufixo *lip20201*, tudo minúsculo. É nele que devem ser colocados os arquivos. Convide o professor (jmarcelo.alencar@gmail.com) para ser colaborador do repositório. Faz parte da avaliação utilizar o GitHub de forma correta. Também faz parte da avaliação descobrir como cronometrar tempo de execução de trechos de código em várias linguagens (veja mais nos enunciados).

Preparação do Diretório.

No repositório criado, crie um diretório chamado *prova04* e dentro dele subdiretórios chamados *questaoXX*, no qual XX deve ser 01 para o subdiretório da primeira questão, 02 para o subdiretório da segunda questão e assim por diante. É nele que você irá colocar os arquivos de cada questão. Obedeça ao formato, usando minúsculas e evitando espaços ou outros separadores. Falha em obedecer às regras de nomenclatura acarretam anulação automática.

Questões

Questão 01 – Exercícios de Programação 05, Capítulo 04, página 225.

Nome do arquivo deve ser *questao01.c*. Modifique o analisador léxico dado na Seção 4.2 para reconhecer a seguinte lista de palavras reservadas e retorne seus códigos de *token* respectivos: **for** (FOR_ CODE, 30), **if** (IF_CODE, 31), **else** (ELSE_CODE, 32), **while** (WHILE_CODE, 33), **do** (DO_CODE, 34), **int** (INT_CODE, 35), **float** (FLOAT_CODE, 36), **switch** (SWITCH_CODE, 37).

Questão 02 – Exercícios de Programação 03, Capítulo 05, página 266.

Nome do arquivo deve ser *questao02.js.* Escreva um *script* em *JavaScript* ou *NodeJS* que tenha subprogramas aninhados em três níveis de profundidade e nos quais cada subprograma aninhado referência variáveis definidas em todos os seus subprogramas que o envolvem no aninhamento.

Questão 03 – Exercícios de Programação 06, Capítulo 05, página 266.

Nome do arquivo deve ser *questao03.c* ou *questao03.cpp*. Escreva três funções em C ou C++: uma que declare um grande vetor estaticamente, outra que declare o mesmo grande vetor na pilha e outra que crie o mesmo grande vetor no monte. Chame cada um desses subprogramas muitas vezes (ao menos 100 mil vezes) e mostre na tela o tempo necessário para cada um. Pesquise no Google como fazer a medição de um trecho de código em C, você também será avaliado por sua capacidade de escolher uma boa técnica. Em um comentário de várias no começo do arquivo, escreva um parágrafo ou dois explicando as diferenças.

Questão 04 – Exercícios de Programação 05, Capítulo 06, página 340.

Nome do arquivo deve ser *questao04.cpp* e *questao04.cs.* Escreva um programa simples em C++ para investigar a segurança de seus tipos de enumeração. Inclua ao menos 10 operações diferentes em tipos de enumeração para determinar se coisas incorretas ou bobas são legais. Agora, escreva um programa em C# que faça as mesmas coisas e execute-o para determinar quantas dessas coisas incorretas ou bobas são legais. Compare seus resultados explicando em um comentário de várias linhas no começo do arquivo *questao04.cpp* o que é possível em uma linguagem, mas impossível em outra.

Questão 05 – Exercícios de Programação 07, Capítulo 06, página 340.

Nome do arquivo deve ser *questao05.c.* Escreva um programa em C que faça muitas referências para elementos de matrizes de duas dimensões, usando apenas índices. Escreva um segundo programa que faça as mesmas operações, mas que use ponteiros e aritmética de ponteiros para a função de mapeamento de armazenamento para fazer as referências aos elementos da matriz. Compare a eficiência em termos de tempo dos dois programas, mostrando qual dos dois é mais rápido. Qual dos dois é mais confiável? Por quê? Responda em um comentário de várias linhas no começo do arquivo.

Questão 06 – Exercícios de Programação 01 e 02, Capítulo 07, página 369.

Os nomes dos arquivos devem ser *questao06.c, questao06.cpp, Questao06.java* e *Questao06.cs*. Rode o código dado no Problema 13 (no Conjunto de Problemas do Capítulo 07) em algum sistema que suporte C para determinar os valores de *sum1* e *sum2*. Explique os resultados em um comentário de várias linhas no começo do arquivo *questao06.c*. Reescreva o programa do em C++, Java e C#, execute-os e compare os resultados, afirmando ainda no começo do arquivo *questao06.c* quais os valores de *sum1* e *sum2* em cada linguagem.

Questão 07 – Exercícios de Programação 05, Capítulo 08, página 411.

Os nomes dos arquivos devem ser *questao07.c, questao07.cpp, Questao07.java* e *Questao07.cs*. Em uma carta para o editor da CACM, *Rubin* (1987) usa o seguinte segmento de código como evidência de que a legibilidade de algum código com *gotos* é melhor do que o código equivalente sem *gotos*. Esse código encontra a primeira linha de uma matriz inteira *n* por *n* chamada *x* que não tem nada além de valores iguais a zero.

```
for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++)
                if (x[i][j] != 0)
                        goto reject;
        println ('First all-zero row is:', i);
```

```
        break;
reject:
}
```

Reescreva esse código sem gotos em uma em cada uma das seguintes linguagens: C, C++, Java, C# ou Ada.

Questão 08 – Exercícios de Programação 08 e 09, Capítulo 08, página 412.

Os nomes dos arquivos devem ser *questao08.c Questao08*.java. Reescreva o segmento de programa em C do Exercício de Programação 4 usando sentenças *if* e *goto* em C. Reescreva o segmento de programa em C do Exercício de Programação 4 em Java sem usar uma construção *switch.*

Questão 09 – Exercícios de Programação 02, Capítulo 09, página 469.

O nome do arquivo deve ser *questao09.xxx*, onde *xxx* depende da linguagem que você escolher. Escreva um programa em uma linguagem que você conheça para determinar a taxa de tempo necessária para passar uma grande matriz por referência e por valor. Faça com que a matriz seja o maior possível na máquina e na implementação que você usa. Passe a matriz tantas vezes quantas forem necessárias para obter tempos razoavelmente precisos das operações de passagem. A questão só irá valer se realmente for demonstrado uma diferença de tempo na execução do subprograma com passagem por referência em relação ao subprograma com passagem por valor.

Questão 10 – Exercícios de Programação 06, Capítulo 09, página 469.

O nome do arquivo deve ser *10.xxx*, onde *xxx* depende da linguagem que você escolher. Escreva um programa em alguma linguagem que tenha tanto variáveis locais estáticas quanto dinâmicas da pilha em subprogramas. Crie seis grandes matrizes (ao menos 100 por 100) no subprograma – três estáticas e três dinâmicas da pilha. Preencha duas das matrizes estáticas e duas das matrizes dinâmicas da pilha com números aleatórios na faixa de 1 a 100. O código no subprograma deve realizar muitas operações de multiplicação de matrizes nas matrizes estáticas e cronometrar o processo. Então, ele deve repetir esse processo com as matrizes dinâmicas da pilha. Compare e explique os resultados em um comentário de várias linhas no começo do arquivo.