UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

SSC0603 – Estrutura de Dados I – 2024-2 (ED1)

Professor responsável: *Fernando Santos Osório*
Semestre: 2024/2
Horário: Quinta 13h/14h (Prova P1)

Wiki: SSC-603-2024(FOsorio)
Web: ***http://www.icmc.usp.br/~fosorio /***
DATA PROVA: 10/10/2024 - PROVA P1

NRO. USP: **< Colocar o seu NUSP no programa fonte>** COMO COMENTÁRIO no CÓDIGO

NOME : **< Colocar o seu Nome no programa fonte>** COMO COMENTÁRIO no CÓDIGO

>> COLOCAR SEU NOME E NRO. USP COMO COMENTÁRIO DO PROGRAMA FONTE ENTREGUE!

PROVA PROVA P1 – SSC0603 – ED1

Q1 (Questão única). Implemente o seguinte programa descrito abaixo de acordo com o especificado. O programa deve ser enviado para o RunCodes da Disciplina SSC0603 (Prova P1)
RunCodes SSC0603 - https://runcodes.icmc.usp.br/offerings/view/62 - Regras da Prova no final deste texto.

O programa da prova é baseado na "famosa" CALCULADORA INFINITA do prof. F.Osório. Deve ser somados uma sequência de dígitos (lidos de 2 arquivos) e exibido o resultado da soma, USANDO LISTAS ENCADEADAS DINÂMICAS (TAD visto em aula ou outro qualquer desde que tenha alocação dinâmica de nodos e a lista seja encadeada com ponteiros).

Descrição: Criar um programa "modular" (com sub-rotinas, conforme descrição mais abaixo), que realize as seguintes tarefas:

(1) Ler dois arquivos texto de entrada que possuem os dígitos de um "número (quase) infinito". Os arquivos são denominados de "nro1.txt" e "nro2.txt", com os dígitos que terão que ser "alinhados" e somados. Os arquivos têm 1 dígito em cada linha, terminando com uma letra 'I' (se forem dígitos da **parte inteira** – ambos arquivos serão do tipo 'I' com a parte inteira) ou com uma letra'D' (se forem dígitos da **parte decimal** – e neste caso também ambos os arquivos serão do tipo 'D' com a parte decimal). A soma é feita parte inteira com parte inteira, OU, parte decimal com parte decimal. Veja os exemplos de arquivos abaixo:

nro1.txt	nro2.txt
1	
2	1
3	2
4	3
5	4
I	I

12345 + 01234 = 13579

nro1.txt	nro2.txt
4	1
3	1
2	1
1	1
D	1
	D

0.43210 + 0.11111 = 0.54321

[Nota importante] Esta prova vai ter uma avaliação maior, de 8.0 pts, se o programa conseguir fazer corretamente pelo menos a soma INTEIRA (da parte inteira), e o restante de 2.0 pts, se também fizer corretamente a soma DECIMAL (da parte decimal). O programa faz uma ou outra soma dependendo da letra final do arquivo que indica se é uma soma do tipo INTEIRO 'I' maiúsculo ou do tipo DECIMA 'D' maiúsculo. Os arquivos sempre serão "nro1.txt" e "nro2.txt".

(2) Você deve ler os dígitos e criar uma lista encadeada com cada número. Assim, poderemos ter um "calculadora infinita" capaz de somar tantos dígitos quanto se desejar. Exibir na tela (printf) o nro1 lido e na linha seguinte o nro2 lido.

(3) Você deve somar os dígitos **1-por-1 alinhados**, **considerando o VAI-UM**, que passa de um dígito (nodo) para o próximo (nodo seguinte). Mas antes, faça o "alinhamento" dos nodos. Atenção:

- Cuidado com o dígito MAIS SIGNIFICATIVO (mais a esquerda) e o MENOS SIGNIFICATIVO (mais a direita). No caso de um número inteiro, o primeiro dígito do arquivo é o mais significativo e o último é o menos significativo. Exemplo nro1.txt (tipo 'I'): 12345 (1 é o mais significativo e o 5 é o menos significativo).
- No caso da soma de um **número inteiro**, começamos somando do número MENOS SIGNIFICATIVOS e seguindo a soma na direção do MAIS SIGNIFICATIVO, lembrando que toda soma de 2 dígitos que for maior ou igual a 10, leva o VAI UM para o dígito seguinte (por isso é importante estarem alinhados!). Este VAI UM é "ativado" e "desativado" conforme o resultado da soma, e deve ser somado na casa seguinte (e verificado novamente).
- Você deve ALINHAR os números INTEIROS ('I') pelo dígito MENOS SIGNIFICATIVO, pois os números podem ter uma quantidade diferente de dígitos. SUGESTÃO: Adicione zeros na parte mais significativa do número menor, de modo a deixar ambos números com a mesma quantidade de dígitos. Exemplo:

  123456 + 12 => 123456 + **000012** => 123467 (Foram colocados zeros para facilitar o cálculo)

Se for feita a **soma das casas decimais ('D'),** o alinhamento é diferente:

- Você deve ALINHAR os números DECIMAIS ('D') adicionando, se necessário, dígitos MENOS SIGNIFICATIVO, pois os números podem ter uma quantidade diferente de dígitos. SUGESTÃO: Adicione zeros na parte menos significativa do número menor de modo a deixar ambos números com a mesma quantidade de dígitos. Exemplo:

  0.123456 + 0.12 => 0.123456 + 0.120000 => 0.243467 (Foram colocados zeros para facilitar)

  Alinhamos adicionando zeros na direita.

(4) Uma vez lidos os arquivos, criadas as 2 listas de dígitos, realizado o alinhamento dos dígitos (nodos), deve ser realizada a soma dos dígitos, 1-por-1, considerando o vai-um, e gerando uma terceira lista encadeada com o resultado da soma (inserir os nodos que representam a soma dos dígitos, um a um, gerando uma nova lista de dígitos). Lembre-se de:

- Somar os dígitos **1-por-1 alinhados**, **considerando o VAI-UM**, que passa de um dígito (nodo) para o próximo (nodo seguinte). O vai um ocorre quando "estouramos a soma de um dígito", gerando 2 dígitos (por exemplo, 1+9 =10). Você deve garantir que os dígitos fiquem entre 0 e 9, que o vai-um seja considerado na soma, e que o vai um seja "ativado e desativado" a cada soma de dígitos considerados.
- Deve ser implementada uma rotina:
  int soma_digitos_inteiros (Lista* li1,Lista* li2,Lista* li3) // Retorna o vai um
  int soma_digitos_decimais (Lista* li1,Lista* li2,Lista* li3) // Retorna o vai um

  ATENÇÃO: Ao somar 2 números se houver um vai um, no dígito mais significativo, NÃO DEVEMOS CRIAR um NOVO NODO, MAS SIM, INDICAR QUE TEVE UM "VAI-UM" FINAL NA SOMA (retorno). Exemplo:
  911 + 100 = 011 e retorna 1
  0.911 + 100 = 0.011 e retorna 1 // O retorno do vai um permite integrar parte inteira e decimal!
  // E evita de "aumentar o nro. de nodos original dos números dados

(5) Exibir na tela (printf) o número final obtido na soma (3ª. lista), seguido de uma nova linha e valor do vai-um (0 ou 1) com uma nova linha.

- Portanto, o programa lê dois arquivos, exibe os números lidos na tela (1 em cada linha – antes de fazer o alinhamento), criando listas encadeadas, realiza a soma (conforme o tipo 'I' ou 'D") criando uma nova lista, e no final exibe na tela o resultado da soma, seguido pelo "vai-um" final da soma. Exemplos de saída na tela:

Entrada:	Entrada:
Nro1 = 100 Nro.2 = 99 (tipo 'I")	Nro1 = 100 Nro.2 = 911 (tipo 'I")
Saída:	Saida:
100	100
99	911
199	011
0	1

BOA PROVA!!!

REGRAS EM RELAÇÃO REALIZAÇÃO DESTA PROVA

1. A PROVA É **INDIVIDUAL com consulta Papel (Livros, papel impresso ou escrito) e formato Digital (Internet, Wiki, Pendrive), porém SEM CONSULTAR ou SE COMUNICAR COM HUMANOS ou QUALQUER OUTRA FORMA DE VIDA TERRESTRE ou EXTRA-TERRESTRE!**
   Não podem usar de formas de comunicação com pessoas externas, além do professor, referente a prova, seja por celular (manter desligado/guardado), por e-mail, por whatsapp, por mensagens ou fóruns ("ao vivo"), com colegas, etc.
   NÃO É PERMITIDO O EMPRÉSTIMO DE MATERIAL (Cadernos, Anotações, Livros, PenDrive, etc).
2. A **PROVA TEM QUE SER FEITA NO LABORATÓRIO 8.113 / 8.115 USANDO OS COMPUTADORES PCs DO LABORATÓRIO (SEM VPN, SEM ACESSO REMOTO!)**
   O ENDEREÇO DE IP USADO (Endereço de rede) PARA O ENVIO DA PROVA TEM QUE SER DAS MÁQUINAS DO LABORATÓRIO 8.113/8.115
   >> As máquinas do Lab. 8.113 e 8.115 estão sendo monitoradas << Sniff, sniff 😉
3. A PROVA DEVE SER **ENTREGUE via RunCodes.icmc (pode submeter várias vezes)**
   ENVIAR O ARQUIVO ZIP do MAKEFILE – O envio de ".c" é "aceito", mas tem *desconto* na nota.
   Entrega é via https://runcodes.icmc.usp.br/ Disciplina SSC0603 – Prova P1
4. **EM CASO DE PROBLEMAS no RunCodes.icmc pode ser entregue por E-Mail** para:
   fosorio @ icmc.usp.br com **cópia** (**Cc:**) para fosorio @ gmail.com
   Com assunto/subject: **Prova-P1 <Seu_Nome> <Nro_USP>**
   ANEXANDO O PROGRAMA FONTE (Zip do Makefile ou se não conseguir, o ".C" e o Project ".CBP")
   NÃO ANEXAR EXECUTÁVEIS! JAMAIS ENVIE EXECUTÁVEIS POR E-MAIL
   NÃO ANEXAR OBJ, BIN e EXE!!! Seu e-mail não será recebido se for anexado bin, obj ou exe!
5. A PROVA USUALMENTE DEVE SER REALIZADA USANDO O CODEBLOCKS DO LABORATÓRIO, e depois enviada (testada com os casos de teste) do RunCodes (USUAL).
6. **AO ENTREGAR A PROVA NO RUNCODES ou POR E-MAIL O(A) ALUNO(A) CONCORDA COM ESTAS REGRAS E SE COMPROMETE A FAZER A PROVA INDIVIDUALMENTE!**

FIM