Universidade Federal do ABC
3º quadrimestre de 2023

Professor: Maycon Sambinelli

Programação Estruturada
PROJETO FINAL

Erick Augusto Silva Pereira — 11202130310

João Vitor Trindade Oliveira da Costa — 11202130604

Vitor Malavasi Silva — 11202020929

Relatório do Projeto "*BigNumber*":

Para a realização do projeto de uma calculadora de *BigNumbers* (números muito grandes, com centenas de dígitos, por exemplo) em C, foram criados os arquivos:

- "bignumber.h" - um *header*, com a interface pública do nosso projeto;
- "bignumber.c" - com a implementação da interface pública;
- "client.c" - com a função "main()" do projeto utilizando a biblioteca "bignumber.h"; e
- "makefile" - para compilar conjuntamente os arquivos do projeto.

O registro, ou *struct*, criado para armazenar os *BigNumbers* é composto por:

- um caractere "*sign*", com o sinal do número ('+' para positivo ou '−' para negativo);
- um vetor de caracteres "*digits*" para armazenar os algarismos dos *BigNumbers*; e
- um inteiro "*size*", com o tamanho dos *BigNumbers*.

Primeiro, foram criadas funções para a leitura de vetores que podem conter infinitos dígitos (limitados apenas pela memória do computador) com alocação dinâmica e para a leitura do sinal da operação matemática a ser realizada ('+', '−' ou '*') e, em seguida, uma função para a inicialização dos *BigNumbers* utilizando as funções de leitura e uma para a liberação de memória alocada para os números.

Os dígitos dos números são armazenados como caracteres, portanto, foi criada uma função que os transforma em inteiros. Entretanto, como os dígitos vão somente de 0 a 9, continuamos a usar o *data type* "*char*" para armazená-los, uma vez que ocupam somente 1 byte, ao invés dos 4 ocupados por variáveis do tipo "*int*". A função que transforma os símbolos dos números nos números decimais em si somente subtrai o decimal 48 das variáveis de dígitos, uma vez que o '0' equivale ao decimal 48 na Tabela ASCII.

Também criamos funções para inverter os números, com a casa da unidade no início, seguida da dezena, da centena, e assim por diante — isso faz com que as contas (soma, subtração e multiplicação) sejam mais simples de serem realizadas, uma vez que serão feitas dígito por dígito partindo das unidades. E uma função para adicionar um "zero à esquerda" aos números.

Percebemos que seria prática a criação de funções para que pudéssemos comparar dois *BigNumbers* quanto a seus tamanhos (quantidade de dígitos) e seus valores, pois essa informação é necessária na hora de somar ou subtrair dois números.

Por fim, montamos as funções que fariam, efetivamente, as operações aritméticas (uma para soma, uma para subtração e uma para multiplicação) e uma função "gerenciadora" que analisa quando cada uma deve ser feita (além de apenas chamar a função de soma quando o caractere operador é '+', ela analisa também os sinais dos *BigNumbers* em questão — a soma entre um número positivo e um negativo, por exemplo, pode ser calculada como a subtração entre dois números positivos, ou seja, o módulo desses números). Para a soma e subtração, fizemos com que um dos números da operação fosse substituído pelo resultado, dígito a dígito (enquanto a conta é feita), para que não fosse necessária a alocação de um terceiro vetor "resultado".

A princípio, montamos o projeto localmente no computador, mas, ao aprendermos sobre a praticidade do uso do *git*, passamos a hospedar o repositório no GitHub.

Acreditamos que nosso projeto tenha ficado bem estruturado e, na medida do possível, simples de ser lido e analisado. Ao longo de todo o arquivo da interface pública do programa e algumas vezes no arquivo com a função "main()", adicionamos comentários com explicações sobre a função ou a linha de código à qual os mesmos se referem.