PROJETO FINAL DE PROGRAMAÇÃO ESTRUTURADA - “BIG NUMBERS”

Arthur Dae Sung Han 11202232206

Guilherme Meira Rebello 11202130848

Isabele Cristina Felix Santos 11202020943

Link Github: https://github.com/daegiri/bignumber

Para o desenvolvimento do projeto, criamos uma estrutura específica para a representação dos BigNumbers, utilizando sempre que possível alocação de memória dinâmica e destruindo essa memória após utilização, estes são compostos pelos dígitos inseridos pelo usuário, pelo comprimento e também por uma estrutura booleana que armazena se o número é ou não negativo. Além disso, optamos por interpretá-los como strings, uma vez que estaríamos lidando com números positivos e negativos, algumas funções criadas que são usadas durante este processo de leitura e criação do big number são:

*read_input(int *size)*

*create_big_number(char *digits, int size, bool is_negative)*

read_big_number()

*destroy_big_number(struct big_number *number)*

Os cálculos são realizados em processos e as operações foram divididas em diferentes funções para cada objetivo, algumas funções relacionadas são:

*add_process(struct process *process, struct expression *expression)*

*destroy_process(struct process *process)*

*print_process(struct process *process)*

Não utilizamos nenhum mecanismo alternativo para otimizar o processo devido ao prazo, preferimos focar na nossa estrutura autoral e na correção de todos os erros para maior eficiência durante os testes e, portanto, um próximo passo para este projeto seria testar outras estruturas de dados avançadas e talvez diferentes algoritmos para obter o processamento mais rápido possível.

A divisão de tarefas foi feita da seguinte maneira: Arthur ficou responsável pela estrutura de dados base, estruturou a primeira função de adição com sinais positivos, fez validação de testes e alguns ajustes finos, Guilherme estruturou a função de multiplicação completa, com ambos os sinais, e também fez alguns outros ajustes finos no processo e Isabele estruturou as funções de adição e subtração completas, com sinais iguais e sinais opostos, fez o levantamento de dados e o mapeamento da estrutura do projeto com a inclusão de comentários, assim como redigiu o readme.

Para compilar o arquivo deve-se utilizar o comando “make”, para rodar o código é possível optar por digitar ./client no terminal ou ./automake.sh e, para finalizar, utilizar o atalho ctrl+D no teclado. Desenvolvemos também scripts de testes automatizados que podem ser executados com o comando ./autotest.sh.