MAC 2166 – Introdução à Ciência da Computação
Grande área Elétrica – Primeiro Semestre de 2021

Terceiro Exercício-Programa — Numerão

> Existem muitas versões para a origem do nome Google, a famosa empresa norte-americana que se destaca, entre outras coisas, pelos seus serviços de busca na Web. Uma das explicações mais aceitas vem de fontes ligadas à Universidade de Stanford, local onde tal empresa foi criada. Essa versão da história conta que o nome pretendido era na verdade "Googol" e um dos seus fundadores cometeu um erro de grafia e acabou escrevendo "Google". Googol é o nome dado ao número 10^{100}. A ideia dos fundadores era escolher um nome que refletisse a imensa quantidade de informação que precisaria ser processada para indexar as páginas da Internet. Neste exercício-programa você contribuirá com a análise grandes números fornecendo um programa para realizar a soma de inteiros muito grandes.

Sobre a invenção do nome "Google", isso poderia muito bem ter sido feito pelos zimbabuanos. Existe uma lenda na Internet que diz que a inflação anual estimada em 2008 teria sido por volta de $6,5 \times 10^{108}\%$ ou 650 milhões de googols (googois?). Essa lenda surgiu do equívoco de alguém ter usado a inflação estimada de janeiro a novembro (79.600.000.000%) como sendo a inflação apenas do mês de novembro e partir daí extrapolando a estimativa anual[1]. Mas seria muito legal ter uma nota de Cem Trilhões de Dólares (ainda que sejam dólares zimbabuanos).

Figura 1: Nota de Z$100.000.000.000.000.

Para se ter uma ideia do quão absurdamente grande é um googol, veja a sua comparação com outros números enormes[2]:

- Desde que ocorreu o Big Bang, "só" se passaram 17×10^{39} de ioctossegundos (1 ioctossegundo = 10^{-24} segundo).
- Juntas, todas as pessoas do mundo viveram 5×10^{11} anos, ou 17×10^{18} segundos, ou "apenas" 17×10^{42} ioctossegundos.
- A massa do universo observável é estimada entre 10^{50} e 10^{60} Kg.

[1] `https://en.wikipedia.org/wiki/Hyperinflation_in_Zimbabwe`
[2] Exemplos "emprestados" da Wikipedia: `https://pt.wikipedia.org/wiki/Googol`.

- Um Googol é aproximadamente igual a 70! (70 fatorial).

Tendo em vista este tipo de problema, é comum em sistemas computacionais precisarmos de estruturas de dados que comportem números gigantescos. A maneira mais comum é guardar estes números em variáveis do tipo *float* que, no final das contas (sem trocadilhos), armazenam uma aproximação do número em notação científica. Isso funciona em grande parte dos casos. Contudo, em alguns ramos da computação, como a criptografia, é preciso calcular e manipular números extremamente grandes (da ordem de 10^{300} ou mais) precisamente, sem aproximações. Comumente, variáveis do tipo inteiro em C vão apenas até 64 bits (ou $2^{64} \sim= 1,8 \times 10^{19}$) e alguns compiladores mais recentes têm suporte a inteiros de até 128 bits ($2^{128} \sim= 3,4 \times 10^{38}$) mesmo em máquinas 64 bits.

Neste projeto queremos ir além. Nós vamos desenvolver uma estrutura de dados capaz de armazenar números muito grandes, superando a limitação do tipo utilizado em C. Neste exercício-programa seu programa será capaz de lidar com números inteiros grandes, sobre os quais efetuaremos a operação de soma (+).

1 Organização do código

Para facilitar nossa discussão, vamos carinhosamente chamar um número a ser armazenado e manipulado por nós de "numerão".

1.1 Representação

Existem maneiras muito elaboradas e eficientes para representar numerões em computadores. Uma das mais empregadas (senão a mais) chama-se *complemento de dois*. Apesar da sua eficiência, sugerimos que neste trabalho você fique longe desta técnica e aborde o problema de uma maneira muito mais simples.

Nossa sugestão é que você armazene os dígitos do numerão em um vetor de inteiros. Assim, o numerão 12345 poderia ser representado pelo vetor $[5, 4, 3, 2, 1]$. Note que sugerimos que no vetor a ordem dos dígitos seja dada dos dígitos menos significativos para os mais significativos. Isso facilita na hora de fazer a soma, por exemplo, pois o algoritmo da soma tradicional (papel e lápis) trabalha dos dígitos menos significativos para os mais significativos.

Será necessário escolher uma maneira de representar números negativos e positivos. Duas maneiras simples de fazê-lo são:

- Fazer o sinal do numerão ser o mesmo do último dígito. Por exemplo: $-310 \rightarrow [0, 1, -3]$, $247 \rightarrow [7, 4, 2]$.
- Reservar uma posição no vetor para indicar o seu sinal, por exemplo a posição 0. Caso contenha 0 o número é positivo, caso contenha 1 é negativo. Exemplo: $-310 \rightarrow [1, 0, 1, 3]$, $247 \rightarrow [0, 7, 4, 2]$.

Armazenando o numerão em um vetor de inteiros, você deve desenvolver as seguintes funções para criar um numerão a partir de um inteiro (`int n`) e para imprimir o numerão armazenado num vetor (`int num[]`):

```
/* Função que recebe um inteiro n e o armazena em num,
retornando a quantidade de dígitos: */
int criaNumerao(int n, int num[]);


/* Função que recebe um numerão com tamNum dígitos e o imprime: */
void imprimeNumerao(int num[], int tamNum);
```

1.2 Operações

Será necessário criar uma função para calcular a soma de numerões. Faça para a soma uma função que altere um dos dois numerões envolvidos na operação ao invés de ter um novo numerão apenas para o resultado da operação.

```
/* Soma 'a' e 'b' e guarda o resultado (a+b) em 'a'
   'b' não é modificado
   retorna a quantidade de dígitos do resultado */
int soma(int a[], int tamA, int b[], int tamB);
```

Para a implementação da operação propriamente dita, utilize o algoritmo para soma com que você está acostumado (papel e lápis). Ele é surpreendentemente eficiente, mesmo para números extremamente grandes. Você pode criar outras funções que julgar oportunas.

Sugestão para o desenvolvimento da soma de inteiros a e b:

- Defina uma constante TAM_MAX, com valor alto (por exemplo, #define TAM_MAX 2021) para ser o tamanho dos vetores de inteiros criados no programa.
- Comece trabalhando apenas com números positivos.
- Implemente a operação de soma.
- Tendo feito o caso em que $a \geq 0$ e $b \geq 0$, resta cuidar dos casos restantes, dependendo se a e b são positivos ou negativos: (i) $a \geq 0$ e $b < 0$; (ii) $a < 0$ e $b \geq 0$; e (iii) $a < 0$ e $b < 0$.

2 Entradas e saídas

Seu programa deve fornecer duas opções para o usuário, inicialmente imprimindo a seguinte mensagem:

```
Digite 0 (soma) ou 1 (soma naturais):
```

Feito isso, um número é lido para armazenar 0 ou 1, de modo que o usuário deve digitar

- 0 - para ler dois inteiros a e b e imprimir a soma $a + b$ utilizando a função "soma";
- 1 - para ler um natural n e imprimir a soma dos primeiros n naturais.

Seu programa deve se comportar como nos exemplos abaixo.

```
Digite 0 (soma) ou 1 (soma naturais): 0
Digite o primeiro numero: 1234
Digite o segundo numero: -5678
Soma: -4444

Digite 0 (soma) ou 1 (soma naturais): 0
Digite o primeiro numero: 1000
Digite o segundo numero: 987
Soma: 1987

Digite 0 (soma) ou 1 (soma naturais): 0
Digite o primeiro numero: -1234
Digite o segundo numero: -5678
Soma: -6912

Digite 0 (soma) ou 1 (soma naturais): 0
Digite o primeiro numero: -1000
Digite o segundo numero: 987
Soma: -13

Digite 0 (soma) ou 1 (soma naturais): 1
Entre com valor de n para soma dos n primeiros naturais: 100000000
Soma dos 100000000 primeiros naturais = 5000000050000000

Digite 0 (soma) ou 1 (soma naturais): 1
Entre com valor de n para soma dos n primeiros naturais: 4
Soma dos 4 primeiros naturais = 10

Digite 0 (soma) ou 1 (soma naturais): 1
Entre com valor de n para soma dos n primeiros naturais: 123456789
Soma dos 123456789 primeiros naturais = 7620789436823655
```

Para calcular a soma dos n primeiros naturais neste EP, você deve utilizar a função `soma` e efetuar a soma de numerões. Não use nenhum tipo de fórmula para a soma de uma progressão aritmética de modo a efetuar a soma dos n primeiros naturais, o que também enfrentaria problemas para valores grandes de n. De fato, para $n = 123456789$, a soma desejada é 7620789436823655, número que não pode ser armazenado em variável do tipo `int` mas pode ser calculada como soma de numerões.

Bom trabalho a todos!