PRÁTICA 1: Análise Experimental

Implemente em C++ os Algoritmos *prefixMedia1()* e *prefixMedia2()* apresentados a seguir utilizando funções (uma para cada algoritmo). Pede-se:

a) Obtenha os tempos de execução para vários valores distintos (pelo menos 5) de *n* para cada uma das duas funções.

b) Gere um gráfico de *n* x *tempo de execução* para cada um dos algoritmos.

Algoritmo prefixMedia1(*X*,*n*):

Entradas: inteiro positivo *n* e vetor *X* com *n* números

Saída: vetor *A* com *n* números tal que *A*[*i*] é a média dos elementos *X*[*0*],...,*X*[*i*]

```
para i ← 0 até n-1 faça
        a ← 0
        para j ← 0 até i faça
                a ← a+X[j]
        fim para
        A[i] ← a/(i+1)
fim para
retorne A
```

Algoritmo prefixMedia2(*X*,*n*):

Entradas: inteiro positivo *n* e vetor *X* com *n* números

Saída: vetor *A* com *n* números tal que *A*[*i*] é a média dos elementos *X*[*0*],...,*X*[*i*]

s ← 0

para *i* ← 0 **até** *n*-1 **faça**

s ← *s*+*X*[*i*]

A[*i*] ← *s*/(*i*+1)

fim para

retorne *A*

DICA 1: Para obtenção dos tempos experimentais, utilize o comando clock() da biblioteca <ctime>, sendo clock() o tempo do processador em *clocks.* O tempo em *clocks* é de um tipo de dados especial chamado clock_t. Para obter o tempo em segundos deve-se dividir o resultado por CLOCKS_PER_SEC, que é uma constante que fornece o número de *clocks* em um segundo do computador. O fragmento de código a seguir mostra um exemplo.

```
#include <iostream>
#include <ctime>
using namespace std;

int main(){
	clock_t tempo1, tempo2;
	double tempo_total;
	...
	tempo1=clock();
	// inicio da parte do programa em que quer calcular o tempo
	...
	// fim da parte do programa em que quer calcular o tempo
	tempo2=clock();
	// OPCAO 1 PARA CALCULAR O TEMPO
	tempo_total=difftime(tempo2,tempo1)/CLOCKS_PER_SEC;
	cout<<"Tempo="<< tempo_total<<" segundos"<<endl;
	// OPCAO 2 PARA CALCULAR O TEMPO
	cout<<"Tempo="<<1000.0*(tempo2-tempo1)/CLOCKS_PER_SEC<<"
milisegundos"<<endl;
	...
	return 0;
}
```

DICA 2: Utilize alocação dinâmica. Peça o tamanho *n* do vetor e então aloque dinamicamente os vetores **X** e **A** com *n* elementos cada na função int main().

DICA 3: Utilize faixas de *n* diferentes para os dois algoritmos.

DICA 4: Utilize um vetor gerado automaticamente (por exemplo: vetor **X** de uns). Primeiro, teste se a saída está correta, ou seja, se o vetor **A** é correto para um dado vetor **X** com um valor de *n* pequeno. Se estiver correto, então não é preciso mais mostrar o vetor **A** para valores de *n* grandes, apenas os tempos.

DICA 5: Utilize o programa desenvolvido para coletar os tempos para cada valor de *n*. Gere o gráfico utilizando algum aplicativo (por exemplo, Excel) ou à mão.