**Material Maratona de Programação C/C++**

**Introdução:**

**• Rodar código no terminal:**

g++ -lm nomearq.cpp -o a && a.exe (Windows)

g++ -lm nomearq.cpp -o a && ./a.out (Linux)

**• Testar os casos testes:**

g++ -lm nomearq.cpp -o a && a.exe < nomearqentr (Windows)

g++ -lm nomearq.cpp -o a && ./a.out < nomearqentr (Linux)

**• Criar arquivo com respostas para depois comparar com gabarito:**

g++ -lm nomearq.cpp -o a && a.exe < nomearqentr > arqresp (Windows)

g++ -lm nomearq.cpp -o a && ./a.out < nomearqentr > arqresp (Linux)

**• Comparar arquivo de resposta com gabarito no Windows:**

fc arqresp gabarito (Windows)

diff arqresp gabarito (Linux)

**• Biblioteca padrão:**

#include <bits/stdc++.h>

using namespace std;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Sumário**

**Strings**

1. Modos de ler uma string

2. Converter uma string para int

3. Converter int para string

4. Inverter uma string

5. Localizar uma substring

6. Converter string no formato Hexadecimal para Int

7. Retorna o char na posição passado como parâmetro

8. Concatenar uma string na outra

9. Apagar pedaço específico da string

10. Ler palavra por palavra de uma string através de um delimitador

**Ordenação**

1. Passando um vector por referência em uma função

2. Função sort

3. Ordenar vector de struct

**Matemática**

1. Cálculo MDC entre dois números

2. Cálculo MMC entre dois números

3. Juro simples

4. Juro composto

5. Média ponderada

6. Fatorial

7. Arranjo

8. Combinação

9. Fibonacci

10. Permutação simples

11. Permutação com repetição

12. Teorema Fundamental da Contagem

**Outros**

1. Número de casas decimais

2. Converter um inteiro para hexadecimal

3. Ignorar caracteres no scanf

4. Função número primo

5. Números grandes

6. Ano bissexto

7. Arredondar valores

8. Implementação de uma árvore binária / PreOrder / InOrder / PosOrder

9. Procurar elemento em um array/vector

10. Corrigir erro na função stoi

**• *Strings***

**1. Modos de ler uma string**

string s;

cin >> s; // irá ler até o primeiro espaço

getline(cin,s); // irá ler até o \n

scanf(“%[^\n]s”, s); // também irá ler até o \n

**2. Converter uma string para int**

stoi(x);

**Exemplo:**

string s = "32";

int n = stoi(s);

cout << s << “ “ << n << endl;

**Saída:**

32 2

**3. Converter int para string**

Exemplo:

int n = 32;

stringstream ss;

ss << n;

string s = ss.str();

cout << s << endl;

**4. Inverter uma string**

reverse(pos1,pos2);

**Exemplo:**

string s = "Teste String";

reverse(s.begin(),s.end());

cout << s << endl;

**Saída:**

gnirtS etseT

**5. Localizar uma substring**

stringName.substr(pos1,pos2);

**Exemplo:**

string a = "Esse eh somente um teste!";

cout << a.substr(0,a.size()) << endl;

cout << a.substr(1,a.size()-1) << endl;

cout << a.substr(1,a.size()-2) << endl;

**Saída:**

Esse eh somente um teste!

sse eh somente um teste!

sse eh somente um teste

**6. Converter string no formato Hexadecimal para Int**

Podemos usar stringstream nome(nome\_string)

**Exemplo:**

string n = “0x3e8”;

unsigned int = i;

stringstream ss(n);

ss >> hex >> i;

cout << i << endl;

**7. Retorna o char na posição passado como parâmetro**

str.at(value);

**8. Concatenar uma string na outra**

str.append(string);

ou então podemos fazer apenas: str1 += str2;

**9. Apagar pedaço específico da string**

str.erase(pos1,pos2);

**10. Ler palavra por palavra de uma string através de um delimitador**

***String tem que ser assim: char s[];***

strtok(string,delimitador);

**Exemplo:**

char s[] = “RESOLUCAO-DE-PROBLEMAS”;

char \* aux;

aux = strtok(s,”-“);

while(aux != NULL){

cout << aux << endl;

aux = strtok(NULL, “-“);

}

**Saída:**

RESOLUCAO

DE

PROBLEMAS

**• Ordenação**

**1. Passando um vector por referência em uma função**

void function(vector<type> &v){}

Para chamar basta fazer:

vector<int> v;

function(v);

**2. Função sort**

sort(vetor.begin(),vetor.end());

Podemos também ter um terceiro parâmetro na função:

sort(vetor.begin(), vetor.end(), fun);

Func é uma função que irá dizer o modo de ordenar.

**Exemplo:**

bool func(int a, int b){

if(a < b)

return true;

else

return false;

}

vector<int> v = {15,1,5,2};

sort(v.begin(),v.end(),func);

**3. Ordenar vector de struct**

* Criação da struct:

string nome;

string cor;

char tamanho;

**str(){}**

**str(string nome, string cor, char tamanho):nome(nome),cor(cor),tamanho(tamanho){}**

**bool operator < (const str &other) const{**

**if(cor != other.cor){**

**if(cor == "branco") return true;**

**else return false;**

**}else if(tamanho != other.tamanho){**

**if(tamanho == 'P') return true;**

**else if(tamanho == 'M' && other.tamanho == 'G') return true;**

**else return false;**

**}else if(nome != other.nome) return nome < other.nome;**

**}**

* Inserir e ordenar:

**vector<str> v;**

**while(n > 0){**

**string nome, cor;**

**char tamanho;**

**cin.ignore();**

**getline(cin,nome);**

**cin >> cor >> tamanho;**

**v.push\_back(str(nome,cor,tamanho));**

**n--;**

**}**

**sort(v.begin(),v.end());**

**• Matemática**

**1. Cálculo MDC entre dois números**

\_\_gcd(x,y); (gcd = greatest common divisor (máximo divisor comum))

**2. Cálculo MMC entre dois números**

(a / \_\_gcd(a, b)) \* b

Essa é a fórmula para calcular o mmc entre dois números.

**3. Juro simples**

J = C \* i \* t, onde:

J é o juro, C é o capital, i é a taxa e t é o tempo.

**4. Juro composto**

M = C \* (1 + i)^t, onde:

M é o valor final da transação.

**5. Média ponderada**

Leva em conta o peso de cada informação.

**Exemplo:**

Biologia => nota: 8.2, peso: 3

Filosofia => nota: 10.0, peso: 2

Matemática => nota: 6.7, peso: 4

Média Ponderada:

(8.2 \* 3 + 10 \* 2 + 6.7 \* 4) / (3 + 2 + 4)

**6. Fatorial**

5! = 5 \* 4 \* 3 \* 2 \* 1

0! = 1

1! = 1 \* 0! => 1 \* 1 = 1

**7. Arranjo**

Ordem dos elementos irá importar.

**Fórmula: A(n,p) = n! / (n-p)!**

n = qntd de elementos que podem ser escolhidos.

p = qntd de elementos por agrupamento.

**8. Combinação**

Ordem dos elementos não importa.

**Fórmula: C(n,p) = n! / (p! \* (n-p)!)**

* Supondo que queremos ver todas as possibilidades de organizar 8 moedas separando-as de 3 em 3.
* **Com repetição:**

For(i = 0; i < 8; i++)

For(j = 0; j < 8; j++)

For(k = 0; k < 8; k++){

Cout << moeda[i] << [j] << [k] ...

Total++;

}

* **Sem repetição:**

For(i = 0; i < 8-3; i++)

For(j = i+1; j < 8;-2 j++)

For(k = j+1; k < 8-1; k++){

Cout << moeda[i] << [j] << [k] ...

Total++;

Função para pegar todas as permutações possíveis:

**next\_permutation(first,last);**

Exemplo:

int v[] = {1,2,3};

do {

cout << v[0] << " " << v[1] << " " << v[2] << endl;

}while(next\_permutation(v,v+3));

cout << "After loop:" << endl << v[0] << " " << v[1] << " " << v[2] << endl;

**Saída:**

1 2 3

1 3 2

2 1 3

2 3 1

3 1 2

3 2 1

After loop:

1 2 3

**prev\_permutation(first,last);**

* *Esta função precisa passar o vetor em ordem decrescente, enquanto a outra em ordem crescente.*

**is\_permutation**

* Verifica se dois conjuntos é permutação um do outro.

Exemplo:

string s1 = "ABCD";

string s2 = "BADC";

if(is\_permutation(s1.begin(),s1.end(),s2.begin()))

cout << "Is permutation" << endl;

**Saída:**

Is permutation

**9. Fibonacci**

If(num == 1 || num == 2) {

Return 1;

}

Return Fibonacci(num-1) + Fibonacci(num-2);

**10. Permutação simples**

n!

**11. Permutação com repetição**

P = n! / (x! y! z!)

n = nº de elementos

x,y,z = número de repetições de cada elemento

**12. Teorema Fundamental da Contagem**

Tendo 6 camisetas, 4 calças e 2 sapatos. Quantas maneiras de combinação é possível?

**6x4x2 = 48**

**• Outros**

**1. Número de casas decimais**

cout << fixed << setprecision(x);

Onde x é o número de casas decimais.

**2. Converter um inteiro para hexadecimal**

Use o %x para fazer a conversão.

**Exemplo:**

int i = 12;

printf(“%x\n”, i);

*NOTA: SE VC FIZER %x AS LETRAS DO HEXADECIMAL SAEM MINÚSCULAS! SE FIZER %X, ENTÃO AS LETRAS IRÃO SAIR MAIÚSCULAS!*

**3. Ignorar caracteres no scanf**

Supondo que quero digitar duas coisas no scanf.

**Exemplo:**

char \* a;

char \* b;

scanf(“%s%\*c%s”, a,b);

*Esse %\*c irá ignorar algum caracter como ‘ ‘, por exemplo, ou então ‘/’ ou ‘-‘...*

**4. Função número primo**

bool is\_prime(int n){

if(n <= 1)

return false;

int div = 0;

for(int i = 1; i <= sqrt(n); ++i){

if(n % i == 0)

div++;

if(div > 1)

return false;

}

return true;

}

**5. Números grandes**

**Exemplo:**

cout << "Double:\n";

double a = 100000;

double b = 100000;

double c = a \* b;

cout << c << endl;

cout << fixed << c << endl;

**cout << fixed << setprecision(0) << c << endl;**

cout << "\nInt:\n";

int d = 100000;

int e = 100000;

long long int f = d \* e;

cout << f << endl;

**f = d \* 1LL \* e;**

cout << f << endl;

**Saída:**

Double:

1e+010 // a \* b

10000000000.000000 // fixed

10000000000 // fixed and setprecion(0)

Int:

1410065408 // d \* e

10000000000 // d \* 1LL \* e

**Exemplo de problema:**

Given rectangle of length l and b. Print area of rectangle

Constraints:

1 <= l, b <= 10^9

Sample input:

3 4

Sample output:

12

**Cod:**

int l,b;

cin >> l >> b;

long long int area = l \* 1LL \* b;

cout << area << endl;

**6. Ano bissexto**

Para determinar se um ano é bissexto, execute estas etapas:

1. Se o ano for uniformemente divisível por 4, vá para a etapa 2. Caso contrário, vá para a etapa 5.

2. Se o ano for uniformemente divisível por 100, vá para a etapa 3. Caso contrário, vá para a etapa 4.

3. Se o ano for uniformemente divisível por 400, vá para a etapa 4. Caso contrário, vá para a etapa 5.

4. O ano é bissexto (tem 366 dias).

5. O ano não é bissexto.

**Cod:**

bool leap\_year(int y){

if(y % 4 == 0){

if(y % 100 == 0){

if(y % 400 == 0) return true;

else return false;

}else return true;

}else return false;

}

**7. Arredondar valores**

* Para cima – ceil(x)
* Para baixo – floor(x)

**8. Implementação de uma árvore binária / PreOrder / InOrder / PosOrder**

typedef struct Tree{

int value;

Tree\* left;

Tree\* right;

Tree(int value) : value(value), left(nullptr), right(nullptr) {}

}Tree;

Tree\* insertBT(Tree\* tree, int value) {

if(!tree) // árvore vazia

return new Tree(value); // crio um novo nó

if(value > tree->value) { // jogo o elemento maior para a direita da arvore

tree->right = insertBT(tree->right,value);

} else { // jogo o elemento menor para a esquerda da arvore

tree->left = insertBT(tree->left,value);

}

return tree;

}

bool aux; // ajudou printar o problema do modo certo

void preOrder(Tree\* tree) {

if(!tree) return;

if(aux) cout << " ";

cout << tree->value;

aux = true;

preOrder(tree->left);

preOrder(tree->right);

}

void inOrder(Tree\* tree) {

if(!tree) return;

inOrder(tree->left);

if(aux) cout << " ";

cout << tree->value;

aux = true;

inOrder(tree->right);

}

void posOrder(Tree\* tree) {

if(!tree) return;

posOrder(tree->left);

posOrder(tree->right);

if(aux) cout << " ";

cout << tree->value;

aux = true;

}

int main(){

int n;

Tree\* tree = nullptr;

while(cin >> n) {

tree = insertBT(tree,n);

}

aux = false;

preOrder(tree);

cout << endl;

}

**9. Procurar elemento em um array/vector**

Array:

int array[] = {1,5,2};

**int \*p;**

**int sizeArray = sizeof(array)/sizeof(int);**

**p = find(array,array+sizeArray,5);**

**if(p != array+sizeArray)** cout << "Element found in array" << endl;

Vector:

int array[] = {1,5,2};

**int sizeArray = sizeof(array)/sizeof(int);**

**vector<int> v (array,array+sizeArray);**

**vector<int>::iterator it;**

**it = find(v.begin(),v.end(),5);**

**if(it != v.end())** cout << "Elemenet found in vector!" << endl;

**10. Corrigir erro na função stoi**

* c.str()

string s = "123456";

int n = stod(s.**c\_str()**);