Referencia Linguagem C / (algumas) funções do compilador C++
Disciplina Linguagem Científica

1 – Tipos Primitivos

Tipo	Tamanho	Escala	
char	1 byte	-128 a 127	número muito pequeno e caracter ASCII
int	2 bytes	-32768 a 32767	contador, controle de laço
float	4 bytes	3.4e-38 a 3.4e38	real (precisão de 7 dígitos)
double	8 bytes	1.7e-308 a 1.7e308	científico (precisão de 15 dígitos)

unsigned char	1	0 a 255
unsigned int	2	0 a 65 535
long int	4	-2 147 483 648 a 2 147 483 647
unsigned long int	4	0 a 4 294 967 295
long double	10	3.4e-4932 a 1.1e4932

2- Códigos de impressão

%c – caracter
%d – decimal
%e – notação científica
%f – ponto flutuante
%g – ponto flutuante ou notação científica
%o – octal
%s – cadeia de caracteres (string char nome[20])
%u – decimal sem sinal
%x – hexadecimal

2ª-Códigos de leitura

Tipo	define o tipo de dado a ser lido (Requerido)
d	inteiro decimal (int)
f	ponto flutuante (float)
o	inteiro octal (int)
x	inteiro hexadecimal (int)
i	inteiro decimal de qualquer formato(int)
u	inteiro decimal sem sinal (unsigned int)
s	string (char*)
c	caracter (char)

3 – Códigos especiais

Controle/Caracter	Sequencia de escape	Valor ASCII
nulo (null)	\0	00
campainha (bell)	\a	07
retrocesso (backspace)	\b	08
tabulacao horizontal	\t	09
nova linha (new line)	\n	10
tabulacao vertical	\v	11
alimentacao de folha (form feed)	\f	12
retorno de carro(carriage return)	\r	13
aspas (")	\"	34
apostrofo (')	\'	39
interrogacao (?)	\?	63
barra invertida (\)	\\	92

4 – Operadores

\+ - * /
% - resto da divisão
++ - incremento
-- - decremento

var += exp	*var* = var + *exp*
var -= exp	*var* = var - *exp*
var *= exp	*var* = var * *exp*
var /= exp	*var* = var / *exp*
var %= exp	*var* = var % *exp*

+=, -+, *=, /=, %= - operadores aritméticos de atribuição

Operadores Incrementais

++ var	var = var + 1
var ++	var = var + 1
-- var	var = var - 1
var --	var = var - 1

5 –Condicionais / Lógicos

0 corresponde a falso
1 corresponde a verdadeiro

Operador **&&(and):** (op1 && op2)	**op1**	**op2**	**Res**
	1	1	1
	1	0	0
	0	1	0
	0	0	0

Obs: outra forma: (op1 **and** op2)

Operador **||(or):** (op1 || op2)	**op1**	**op2**	**Res**
	1	1	1
	1	0	1
	0	1	1
	0	0	0

Obs: outra forma: (op1 **or** op2)

Operador **!(not):** (!op)	**op**	**Res**
	1	0
	0	1

Operador	**Significado**
>	maior que
<	menor que
>=	maior ou igual a (não menor que)
<=	menor ou igual a (não maior que)
==	igual a
!=	não igual a (diferente de)

6- Comandos de Decisão/Controle – Repetição(Laço)

a)
```
if (teste)
  {comando;}
```

b)
```
if (teste)
  {comandos1;}
else
  {comandos2;}
```

d) switch (expressão constante)
```
{
  case constante1: comando;
                   break;
  case constante2: comando;
                   break;
              :
              :
```

```
   default: comandos;  //opcional
}
```

Comandos de repetição

```
a) while (condição)
      {comando;}

b) do
    {
       comando
     }
    while (condição);

c) for (inicialização; teste; incremento)
        comando;
```

7 – Comandos de Pré- processador – Diretivas de Compilação e "Define"

a) #define nome valor
b) #include <biblioteca>

8 – Funções de E/S

cin>>variável;
cout<<"Texto que sairá no monitor"<<variável...
scanf(%tipo,&variável);
printf("Texo %tipo", variável)
getch() – não se vê o caracter
getche() – se vê o caracter
getchar() – não aceita argumentos/precisa de enter
putchar(valor) – complemento de getchar()
gets() – lê uma string até encontrar fim de linha
puts() – imprime uma string e pula linha
strcmp() – compara 2 strings (considera maiúsculo ≠ minúsculo)
stricmp() – compara 2 strings (considera maiúsculo = minúsculo)
strlen() – retorna tamanho
strcat() – concatena

> É importante que antes ou após leitura de teclado o *buffer* seja limpo através da linha de comando: fflush(stdin);

09 – Modularização - Funções

-Dividem grandes tarefas em tarefas menores.
-É uma unidade de código de programa autônoma desenhada para cumprir uma tarefa particular.

```
tipo nome_funcao (tipo parametro1, tipo parametro2,....)
{
```

```
   declaração das variáveis locais
   corpo da função
}
```

Variáveis locais=> são as variáveis declaradas dentro de uma função e conhecidas somente dentro de seu próprio bloco.

Parâmetros=> são variáveis de comunicação entre a função ativada e a função que a ativa

Passagem de parâmetro Por valor=> significa que à função é dada uma cópia dos valores dos argumentos, e ela cria outras variáveis temporárias para armazenar estes valores. Uma função chamada não pode alterar o valor de uma variável da função que a chamou; ela só pode alterar sua cópia temporária.

Passagem de parâmetro Por Referência=> ao contrário da anterior não é criada uma cópia da variável, para a função é passado o endereço de memória da mesma. Este tipo de função é usado o tipo ponteiro para as variáveis e a função pode então alterar os valor da variável que a chamou.

Tipo da Função
A função deve ser declarada de acordo com o seu valor de retorno. Funções que não retornam valor devem ser declaradas como void. O comando return é usado para efetuar o retorno da função.

10- Estrutura de Dados Vetor - Matriz

Coleção de variáveis do mesmo tipo referenciada por um nome comum. (Herbert Schildt) Características:
Acesso por meio de um índice inteiro.
Posições contíguas na memória.
Tamanho pré-definido.
Índices fora dos limites podem causar comportamento anômalo do programa.

Sintaxe:
<tipo> identificador [<tamanho do vetor>];

Declaração:
```
float notas[100];
 int medias[100];
```

Exemplo:
```
#include<iostream.h>
#include<time.h>
main()
{
  int tam, i, P=0;
  cout<<"Informe tamanho: ";
  cin>>tam;
  int A[tam];
  int B[tam];
```

```
 srand(time(NULL));

 for (i=0; i<tam; i++)
   A[i]=rand()%100;

 for (i=0; i<tam; i++)
   B[i]=rand()%100;

 cout<<"Valores armazenados no vetor A\n\n";
 for (i=0; i<tam; i++)
   cout<<"Vet[ "<<i<<" ] = "<<A[i]<<"\n";
 cout<<"Valores armazenados no vetor B\n\n";
 for (i=0; i<tam; i++)
   cout<<"Vet[ "<<i<<" ] = "<<B[i]<<"\n";

 for(i=0; i<tam; i++)
   P=P + A[i]*B[i];

cout<<"\nProduto Escalar: "<<P;

cout<<"\n\n";
system("pause");

}
```

11 – Operações com Strings

C++ possui uma série de funções e operações próprias para strings. A tabela abaixo resume as operações mais utilizadas (s é uma string qualquer):

- **s.empty( )** - Função que retorna verdadeiro se a string está vazia, e falso caso contrário.
- **s.size ( )** - Função que retorna o tamanho em caracteres da string.
- **s [n]** - Acessa um elemento da string. Funciona exatamente com um elemento de uma matriz.
- **s1 + s2** - Concatena duas strings.
- **s1 = s2** - Atribui o conteúdo de s2 na string s1.
- **s1 == s2** - Testa a igualdade entre s1 e s2 (retorna verdadeiro se as duas strings forem iguais). Duas strings são consideradas iguais se elas tiverem o mesmo número de caracteres e seus caracteres forem iguais.

11.1– Biblioteca cctype: operações com caracteres

A biblioteca cctype é uma versão da biblioteca ctype da linguagem C, convertida para C++. Ela contém diversas funções que permitem processar os caracteres de uma string, um por um. Por exemplo, podemos precisar saber se um determinado caracter é uma letra ou um número, se está acentuado ou não, se é minúsculo ou maiúsculo, e transformar este caractere. Para utilizar esta biblioteca, precisamos declará-la no cabeçalho do programa, assim como fizemos com a biblioteca de strings. Para declará-la, a sintaxe é a seguinte:

```
#include <cctype>
```

Assim, o cabeçalho de um programa que utiliza strings e a biblioteca cctype ficaria assim:

```
#include <iostream>
#include <string>
#include <cctype>
using namespace std;
```

A tabela abaixo resume algumas das funções mais úteis desta biblioteca (x é um elemento de uma string, por exemplo, "sobrenome[4]") :

- **isalnum (x)** - Retorna verdadeiro (1) caso x for uma letra ou um número.
- **isalpha (x)** - Retorna verdadeiro (1) caso x for uma letra.
- **iscntrl (x)** - Retorna verdadeiro (1) caso x for um dígito de controle.
- **isdigit (x)** - Retorna verdadeiro (1) caso x for um número.
- **isgraph (x)** - Retorna verdadeiro (1) caso x não for um espaço.
- **islower (x)** - Retorna verdadeiro (1) caso x for uma letra minúscula.
- **isprint (x)** - Retorna verdadeiro (1) caso x for um caractere imprimível.
- **ispunct (x)** - Retorna verdadeiro (1) caso x for um caractere acentuado.
- **isspace (x)** - Retorna verdadeiro (1) caso x for um espaço em branco.
- **isupper (x)** - Retorna verdadeiro (1) caso x for uma letra maiúscula
- **isxdigit (x)** - Retorna verdadeiro (1) caso x for um número hexadecimal.
- **tolower (x)** - Transforma um caractere maiúsculo em minúsculo.
- **toupper (x)** - Transforma um caractere minúsculo em maiúsculo.

Com exceção das duas últimas funções que transformam caracteres, todas as outras testam os caracteres de uma string, retornando valores booleanos (true ou false, 1 ou 0). O programa abaixo mostra o uso de algumas dessas funções, através da leitura de uma string entrada pelo usuário. Note que também é feito o uso da função <string>.size para determinar o tamanho da string.

12-Estrutura de Dados - STRUCT

As estruturas de dados consistem em criar apenas um dado que contém vários membros, que nada mais são do que outras variáveis. De uma forma mais simples, é como se uma variável tivesse outras variáveis dentro dela. A vantagem em se usar estruturas de dados é que podemos agrupar de forma organizada vários tipos de dados diferentes, por exemplo, dentro de uma estrutura de dados podemos ter juntos tanto um tipo float, um inteiro, um char ou um double.

As variáveis que ficam dentro da estrutura de dados são chamadas de membros.

Declarando a estrutura:

```
1   #include <iostream.h>
2
3   struct Ficha {
4       char  nome[15];
5       char  endereco[20];
6       char  tel[10];
7       float salario;
8       int   idade;
9   };
10
```

Iniciando o main() e criando a variável do tipo estrutura:

```
11  main()
12  {
13      Ficha funcionario;
```

Vetor de estruturas

Usado quando precisamos de diversas cópias de uma estrutura.
Por exemplo: cada cliente de uma locadora de vídeo constitui um elemento de um vetor, cujo tipo é uma estrutura de dados que define as características de cada cliente.
struct nome_da_estrutura variável[dimensão];

Exemplo Problemas reais:

- Coleções de dados que são de tipos diferentes
- Ex.:ficha de um cadastro de cliente

Declarando a estrutura:

```
1   #include <iostream.h>
2
3   struct Ficha {
4       char  nome[15];
5       char  endereco[20];
6       char  tel[10];
7       float salario;
8       int   idade;
9   };
10
```

Iniciando o main() e criando a variável do tipo estrutura:

```
10
11  main()
12  {
13      Ficha funcionario[10];
14      int i;
```

Lendo valores para a variável do tipo estrutura:

```
15
16      cout<<"Informe os dados do funcionario"<<endl;
17      for(i=0;i<2;i++)
18      {
19        cout<<"Nome.....:";
20        cin>>funcionario[i].nome;
21        cout<<"Endereco..:";
22        cin>>funcionario[i].endereco;
23        cout<<"Telefone..:";
24        cin>>funcionario[i].tel;
25        cout<<"Salario...:";
26        cin>>funcionario[i].salario;
27        cout<<"Idade.....:";
28        cin>>funcionario[i].idade;
29        cout<<"\n";
30      }
31      system("pause");
32
```

13 - Exemplos Gerais:

```
#include<iostream.h>
/*diretiva de compilação, permitirá
que o compilador analise o código que
vc digitou apresentando erros de sintaxe, se existirem */

main()//inicio do programa
{  float lado;
/*float para criação de variável
do tipo real (com decimais) */
  cout<<"Digite o lado: ";
/*cout substitui o ESCREVA do
algoritmo - indica a saida de uma
informação na tela */
  cin>>lado;
  fflush(stdin);
/*cin substitui o LEIA do
algoritmo  - indica a leitura de
um valor digitado (teclado)*/
  cout<<"A area = "<<lado*lado<<"\n";
  system("pause");

}

#include<iostream>
using namesapce std:

main()
{ float altura;
  char sexo;

  cout<<"Informe a altura: ";  cin>>altura;
  cout<<"Informe o sexo:   ";  cin>>sexo;
```

```
  if (sexo == 'm' || sexo == 'M')
     cout<<"O Peso ideal = "<<72.7 * altura - 58<<"\n";
  else
     cout<<"O Peso ideal = "<<62.1 * altura - 44.7<<"\n";

system("pause");

}

#include <iostream.h>

main()
{
  int idade;
  cout<<"Informe a idade:  ";
  cin>>idade;
  if ((idade >= 5) &&  (idade<=7) )
    { cout<<"Infantil A";
     cout<<"idade ="<<idade;}
  cin>>idade;
 system("pause");
}

#include <time.h>
#include <iostream.h>
main()
{
int segredo,tentativa=0, num;
srand(time(NULL));//permite a geração de numeros "sem" repetição de valores
segredo=rand()%100; //gera um numero aleatorio de 0 até 100;
do{
   cout<<"Informe sua tentativa: ";
   cin>>num;
   tentativa++;
   if (segredo==num)
   {cout<<"\n Acertou!"<<"em "<<tentativa<<" tentativas"<<"\n";
    cout<<"\nO numero eh = "<<segredo<<"\n";}
    else if (segredo<num)
            cout<<"Errado, muito alto!\n";
         else
            cout<<"Errado, muito baixo!\n";
}while (num!=segredo);

system ("pause");
}

#include <iostream.h>

main ()
{float media, num , qtd_num=0 , soma=0;

cout << "\n" "Digite um valor positivo ou zero ou negativo para sair: " ;
cin >>num;
```

```
while (num>0)
{qtd_num++;
 soma = soma + num;
cout<< "\n" "digite novo valor  positivo ou zero/negativo para sair: " ;
cin >> num;

}
media =(soma/qtd_num);
cout<< " media = " << media<< "\n";
system("pause");
}


#include<iostream.h>
#include<time.h>

int cara_coroa()
{
  int ger;
  ger=rand()%2;
  return(ger);
}

main()
{
 int lanc, cont_cara=0, cont_coroa=0;

 srand(time(NULL));

 for(int i=1;i<=100;i++)
   {  lanc=cara_coroa();
      if (lanc==1)
         cont_cara++;
      else
         cont_coroa++;
}

cout<<"CARA : "<<cont_cara<<"\n";
cout<<"COROA: "<<cont_coroa<<"\n";
system("pause");
}

#include <iostream>
using namespace std;

void trocar (int &a, int &b) {
    int aux;
    aux = a;
    a = b;
    b = aux;
}

 main () {
```

```
   int var1 = 10, var2 = 50;
   cout << "O valor de var1 e " << var1 << endl;
   cout << "O valor de var2 e " << var2 << endl;
   cout<<"\n\n\n";
   trocar (var1, var2);
   cout << "O valor de var1 e " << var1 << endl;
   cout << "O valor de var2 e " << var2 << endl;
   cout<<"\n\n\n";

   system("pause");
}


#include <iostream.h>
using namespace std;

main ()
{float media, num , qtd_num=0 , soma=0;

cout << "\n" "Digite um valor positivo ou zero ou negativo para sair: " ;
cin >>num;
while (num>0)
{ qtd_num++;
  soma = soma + num;
  cout<< "\n" "digite novo valor  positivo ou zero/negativo para sair: " ;
  cin >> num;
}
media =(soma/qtd_num);
cout<< " media = " << media<< "\n";
system("pause");
}

#include<iostream.h>

bool parImpar(int c)
{
    if (c%2==0)
        return true;
    else
        return false;
    }

main()
{
 int a = 12;

 if (parImpar(a)== true)
    cout<<"par";
 else
    cout<<"impar";

 system("pause");
```

```
}


#include <iostream.h>
#include <stdlib.h>
main(){
      int numero;
     cout<<("Digite um numero inteiro: ";
      cin>>numero;
      if (numero%2==0)
        cout<<"\nNumero eh par\n";
      else
        cout<<"\nNumero eh impar\n";
      if (numero>0)
        cout<<"\nNumero eh positivo\n";
      else
        cout<<"\nNumero eh negativo\n";
system("pause");
}

#include<iostream.h>
main()
{

  float vel_motorista, vel_maxima;

      cout<< " informe a  vel_maxima :";
      cin>> vel_maxima ;

      cout<< " informe a  vel_motorista :";
      cin>> vel_motorista ;

      if ((vel_motorista-vel_maxima) > 31)
      {
            cout<< " multa de 200 reais:"<<"\n";
      }
      else
         if ((vel_motorista-vel_maxima) > 10)
         {
           cout<< " multa de 100 reais:"<<"\n";
         }
         else
            {
              cout<< " multa de 50 reais:"<<"\n";
            }
system("pause");

}

#include <iostream>
#include <cstdlib>
#include <cstdio>
#include <strings>
```

```
using namespace std;


#define pula  "\n"


/* */

main()
{
        int num1 = 5, num2 = 3;
        float res;

        setlocale(LC_ALL, "Portuguese");
        cout << "acentuação ";

        cout << "Media direto = " << num1 / num2 << pula;
        res = num1 / num2;
        cout << "Media res sem casting = " << res << pula;
        res = float(num1) / float(num2);
        cout << "Media res casting = " << res << endl;

        system("pause");
}

#include<iostream.h>

main()
{
  int mat[3][3], l, c, soma=0, somal=0, somac=0;
  int somadp=0, somads=0;

  cout<<"Carregando valores na matriz  \n";

  for(l=0; l<3; l++)
    for(c=0; c<3; c++)
        cin>>mat[l][c];

  cout<<"Mostrando valores da matriz \n";
  for(l=0; l<3; l++)
    for(c=0; c<3; c++)
        cout<<"mat [ "<<l<<" ][ "<<c<<" ] = "<<mat[l][c]<<"\n";

  for(l=0; l<3; l++)
    for(c=0; c<3; c++)
       soma = soma + mat[l][c];

 cout<<"A soma = "<<soma<<"\n\n\n";

 for(l=0; l<3; l++)
  {  for(c=0; c<3; c++)
        {
            somal = somal + mat[l][c];
```

```
                    //   0  0
                    //   0  1
                    //   0  2
          somac = somac + mat[c][l];
                    //   0  0
                    //   1  0
                    //   2  0
       }
    cout<<"A soma da linha  "<<l <<"= "<<somal<<"\n";
    cout<<"A soma da coluna "<<l <<"= "<<somac<<"\n";
    somal = somac = 0;
    somadp = somadp + mat[l][l];
                // 0  0
                // 1  1
                // 2  2
   }
    cout<<"A soma da Diagonal Princ = "<<somadp<<"\n";
   c=3;
   for(l=0; l<3; l++)
      { somads = somads + mat[l][c-1-l];
                // 0  3-1-0=2
                // 1  3-1-1=1
                // 2  3-1-2=0

      }
  cout<<"A soma da Diagonal Secundaria = "<<somads<<"\n";
 system("pause");

}

#include <iostream>
#include <string>
using namespace std;

int main()
{
string vazia;
string ditado("Casa de ferreiro, espeto de pau");
string copia_ditado(ditado);
string letra_z( 42, 'z');
cout <<"Mostrando o conteúdo da string 'vazia':"<< endl;
cout << vazia;
cout <<"Mostrando o conteúdo da string 'ditado':"<< endl;
cout << ditado;
cout <<"Mostrando o conteúdo da string 'copia_ditado':"<< endl;
cout << copia_ditado;
cout <<"Mostrando o conteúdo da string 'letra_z':"<< endl;
cout << letra_z;
system("PAUSE > null");
return 0;
}

#include "stdio.h"
```

```
#include "stdlib.h"
#include <iostream.h>

struct ficha{
 char nome [15];
 int data;
 char endereco[20];
 int tel;
 int ano;
} aluno[10];

//struct ficha aluno[10];

int main()
{   int opc, i;


    cin>>opc;

    for(i=1; i<10;i++)
    {
       cout<<"Nome: ";
       cin.getline(aluno[i].nome,15);
       cout<<"Nome: "<<aluno[i].nome;
    }
    system("pause");



/*    int i;
  cout<<"Informe os dados do aluno";
for (i=0;i<2;i++){
    cout<<"Nome:\n";
    cin>>aluno[i].nome;
    cout<<"Data:\n";
    cin>>aluno[i].data;
    cout<<"Endereço:\n"
    cin>>aluno[i].endereco;
    cout<<"Telefone:\n";
    cin>>aluno[i].tel;
    cout<<"Ano que esta cursando:";
    cin>>aluno[i].ano;

}*/

return 0;
}
```