ALGORITMOS E PROGRAMAÇÃO 1º. PER CCOMP – LABORATÓRIO

A.1 Conceito de programa

Um programa é a **implementação de um algoritmo** em uma determinada **linguagem de programação**; é uma sequência de instruções que podem ser executadas por um computador.

A.2 Etapas de processamento de um programa

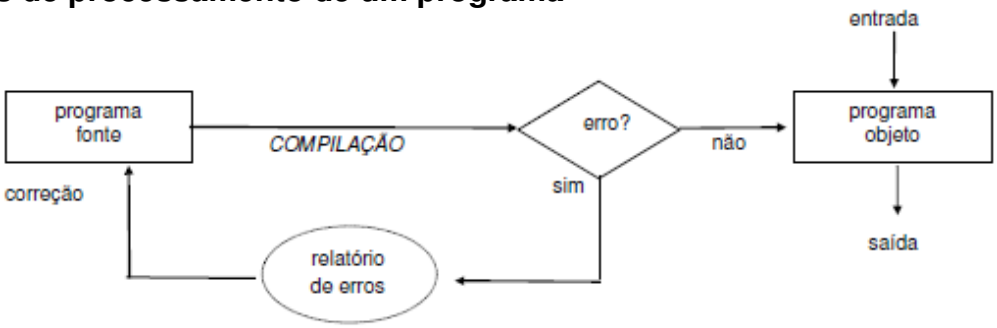
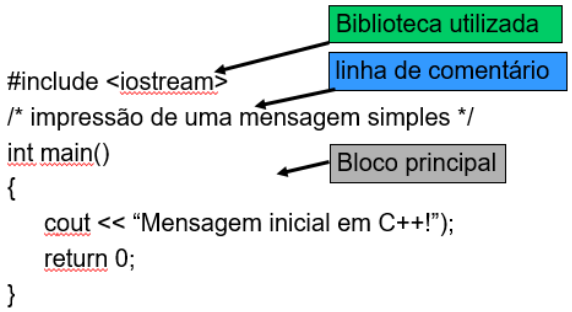


FIGURA 1: etapas de processamento de um programa

A.3 Estrutura dos programas em C++

Os programas escritos em C++ são divididos em várias partes: cabeçalho (comentários, bibliotecas e definições), bloco principal (em geral *int main()*) e blocos auxiliares (classes, rotinas ou funções). Por enquanto usaremos somente as partes.



Bibliotecas são módulos que já estão prontos e disponibilizam funções pré-definidas da linguagem C++ e podem ser utilizados em qualquer programa. Existem inúmeras bibliotecas como: iostream, iomanip, math, ...

A parte do programa definida entre /*...*/ trata-se de um ***comentário de bloco*** e tem como finalidade facilitar a compreensão de um programa. Os comentários podem ser colocados em qualquer ponto do programa. O ***comentário de linha*** é identificado por // e se refere apenas ao texto que se encontra após.
O bloco principal é o corpo do programa, constituído de comandos simples (ATRIBUIÇÃO, ENTRADA DE DADOS e SAÍDA DE DADOS) e comandos estruturados.

Algoritmo	**C++** Tipos e modificadores	**Tamanho** (em bytes, sistemas x86 de 32 bits)	**Intervalo de valores aceitos**
Inteiro	**int**	4 (ou o tamanho nativo do processador)	-2.147.483.648 a 2.147.483.647
	short int	2	-32.768 a 32.767
	unsigned short int	2	0 a 65.535
	unsigned int	4	0 a 65.535 0 a 4.294.967.295
	long int	8	-2.147.483.648 a 2.147.483.647
	unsigned long int	8	0 a 4.294.967.295
Real	**float**	4, com precisão de 7 dígitos	10^{-38} e 10^{38}
	double	8, com precisão de 15 dígitos	10^{-4932} e 10^{4932}
	long double	10	$3.4*10^{-4932}$ a $3.4*10^{4932}$

Algoritmo	C++ Tipos e modificadores	Tamanho (em bytes, sistemas x86 de 32 bits)	Intervalo de valores aceitos
Lógico	bool	1, presente apenas no padrão C99 em diante	true (verdadeiro), false (falso)
Caractere	char	1	-128 a 127
	unsigned char	1, para caracteres individuais do padrão ASCII	0 a 255, ver tabela ASCII
String	string (classe C++)	Tem tamanho variável	Cadeia de caracteres simples

O padrão C de 1999 adicionou um terceiro modificador, suportado pelos compiladores mais recentes, inclusive o gcc, o `long long`, que aumentaria ainda mais a capacidade da variável. Alguns deles suportam esse modificador apenas para o tipo `int`, e outros suportam também para `double`.

Operadores

- **Operadores aritméticos**: + (adição), - (subtração), * (multiplicação), / (divisão), % (resto da divisão inteira MOD), x^y = pow(x,y) (potenciação e radiciação).

  Como há um único operador de divisão (/), o resultado da divisão seguirá a seguinte regra:
  - inteiro / inteiro => inteiro (representando o DIV)
  - real / inteiro ou inteiro / real ou real / real => real (até porque real é o tipo "maior")

- **Operadores relacionais**: == (igual a), != (diferente de), > (maior que), < (menor que), >= (maior ou igual a), <= (menor ou igual a).

- **Operadores lógicos**: ! (não negação), && (e conjunção), || (ou disjunção) (not, and e or também são aceitos)

Comandos

- **comando de atribuição:** variavel **=** valor**;**

  onde valor pode ser um valor fixo (constante), uma outra variável, uma expressão aritmética, relacional ou lógica (combinação de variáveis, constantes e operadores).

- **comandos de entrada de dados – para números:**
  ```
  cin >> variavel;
  cin >> variavel1 >> variavel2 >> ... >> variavelN;
  ```
  Ex.: cin >> idade >> peso; // sendo int idade e float peso

- **comandos de entrada de dados – para caracter simples:**
  ```
  cin >> variavel;
  cin.get ( variavel );  // este é mais adequado
  ```
  Ex.: cin >> letra; // sendo char letra
  cin.get (letra);

- **comando de entrada de dados – para (classe) string:** getline(cin, variavel);

  Ex.: getline (cin, nome); // sendo string nome

- **comando de saída de dados:**
  ```
  cout << variavel;
  cout << texto;
  cout << "texto1" << variavel1 << "texto2" <<... >> endl;
  ```
  onde *endl* faz o cursor pular para a linha abaixo da tela (uso não obrigatório).

  Ex.: cout << " Idade " << idade << " – Peso " << peso;
  cout << " Relatorio" << endl;

- **estruturas de seleção:**

SE-FIMSE
```
if ( teste ) {
   comando(s);
}
```

SE-SENAO-FIMSE
```
if ( teste ) {
   comando(s);
}
else {
   comando(s);
}
```

ESCOLHA
```
switch ( var ) {
  case val1 : comando(s);
              break;
  case val2 : comando(s);
              break;
...
  default : comando(s); // parte optativa
}
```

A.4 Construção de um programa passo-a-passo – CODE::BLOCKS

Para download do ambiente http://www.codeblocks.org/downloads

PASSO no 1: no editor do ambiente, no centro da aba principal clique em New Project OU no via menu clique em File, New, Project. Abrirá para você uma janela para escolher o tipo de projeto (Console application). Clique no botão Go para seguir e escolher a linguagem (C++). Clique em Next para seguir e informar o nome do projeto (Project Title), bem como alterar o diretório se necessário. Clique em Next para seguir e finalizar escolhendo o compilador (GNU GCC Compiler), na sequência clique em Finish. O projeto será criado, e no lado esquerdo inferior da tela (aba Projects), clique no + ao lado de Sources, seguido de duplo clique em main.cpp. Aparecerá um código exemplo na aba central main.cpp, que você deverá alterar, inserindo o código do seu programa.

Digite o programa a seguir:

```
#include <iostream>      // inclusão das bibliotecas necessárias
#include <cmath>

using namespace std;

const float G = 6.67 * pow(10,-11);  // declaração da constante, escopo global

int main() {
      int r12, r23, r13;       // declaração de variáveis, escopo local do main
      float m1, m2, m3, energia;

      // corpo de instruções
      cout << "massas m1, m2, m3 em gramas: "; // informa ao usuario oq ele deve digitar
      cin >> m1 >> m2 >> m3;                // faz a leitura da informacao
      cout << "distancias r12, r13, r23 em m: ";
      cin >> r12 >> r13 >> r23;

      m1 = m1/1000;      // converter gramas para kg em fcao de G
      m2 = m2/1000;
      m3 = m3/1000;
      energia = G * (m1*m2/r12 + m1*m3/r13 + m2*m3/r23) // calculo final

      cout << "Energia de coesao = " << energia << endl; // apresentação do resultado
      return 1;
}
```

PASSO no 2: vamos compilar o seu programa. Você pode agora clicar no botão Build (desenho de uma engrenagem, tela superior a esquerda) e na parte inferior surgirá uma pequena janela de título Build Log mostrando se houve ou não erro de sintaxe no programa (Errors: ?). Caso tenha erros ou *warnings*, volte ao programa para corrigir os erros e compile novamente até não haver mais problemas desta natureza.

PASSO no 3: Com o programa compilado sem erros, a pasta onde foi salvo o programa fonte contém agora o programa executável também. Dentro do ambiente você pode executar o programa clicando em Run (botão ao lado do Build). Deverá surgir uma nova janela, de DOS, mostrando a execução de seu programa.

PASSO no 4: Note que na janela do DOS aparece o tempo de execução do programa e a mensagem “Press any key to continue... “ Depois de conferir o resultado, pressione qualquer tecla para que a janela DOS desapareça. Se a janela DOS não desaparecer, para fechá-la clique no X, no canto superior direito da mesma. Agora que você sabe criar um programa, compilá-lo, salvá-lo e executá-lo, vamos tentar novamente com um programa um pouco mais complexo. Antes de continuar encerre o trabalho com o programa atual. Para tanto clique em File, Close Project.

Observação: propositadamente ocorrerá um erro na compilação deste programa. Corrija após o apontamento do mesmo na compilação !!!!