

Universidade Federal do Rio Grande do Norte
Instituto Metr pole Digital
IMD0030 – Linguagem de Programac o I - T3

Problema: desenvolvimento de habilidades de programac o na linguagem C++.

Subproblema 1: introduc o ao C++, compilac o e depurac o de programas, modularizac o.

Produto do subproblema: (i) documento apresentando as principais caracter sticas e recursos da linguagem C++ que foram identificados durante a explorac o das quest es deste subproblema (em at  duas p ginas, podendo haver ap ndices); e (ii) c digo-fonte dos programas implementados.

Data de entrega via SIGAA: 03 de agosto de 2017.

Instruc es: neste problema o aluno deve consultar as refer ncias indicadas pelo docente para se familiarizar com os recursos necess rios   cria o de programas simples em C++, sem preju zo   consulta de outras fontes como manuais e tutoriais. Usar as quest es e programas mostrados a seguir como guia para as discuss es em grupo e para orientar a explorac o da linguagem C++. Para facilitar o aprendizado, recomenda-se que o aluno compare os recursos e conceitos de C++ com seu conhecimento pr vio acerca de outras linguagens de programac o. Leia e modifique os c digos mostrados e utilize os conceitos e recursos explorados para a criar os programas solicitados. Recursos exclusivos da linguagem C devem ser ignorados e substituídos por seus correspondentes em C++.

Quest es¹:

1. Quais as principais caracter sticas do C++?
2. Podemos afirmar que C++   um super conjunto de C?
3. Quais extens es de arquivo s o comumente utilizadas para identificar c digos-fonte C++?
4. Utilizando o editor de texto Sublime, crie um novo documento a partir da listagem abaixo e, ent o, salve-o com o nome `list0101.cpp`.

```
/// Sort the standard input alphabetically. 1
/// Read lines of text, sort them, and print the results to the standard output. 2
/// If the command line names a file, read from that file. Otherwise, read from 3
```

¹Em grande parte inspiradas em *Exploring C++ 11*, Ray Lischner. Programas retirados desta mesma fonte.

```

4  /// the standard input. The entire input is stored in memory, so don't try
5  /// this with input files that exceed available RAM.
6
7  #include <algorithm>
8  #include <fstream>
9  #include <iostream>
10 #include <iterator>
11 #include <string>
12 #include <vector>
13
14 void read(std::istream& in, std::vector<std::string>& text)
15 {
16     std::string line;
17     while (std::getline(in, line))
18         text.push_back(line);
19 }
20
21 int main(int argc, char* argv[])
22 {
23     // Part 1. Read the entire input into text. If the command line names a file,
24     // read that file. Otherwise, read the standard input.
25     std::vector<std::string> text; ///< Store the lines of text here
26     if (argc < 2)
27         read(std::cin, text);
28     else {
29         std::ifstream in(argv[1]);
30         if (not in)
31         {
32             std::perror(argv[1]);
33             return EXIT_FAILURE;
34         }
35         read(in, text);
36     }
37     // Part 2. Sort the text.
38     std::sort(text.begin(), text.end());
39     // Part 3. Print the sorted text.
40     std::copy(text.begin(), text.end(),
41               std::ostream_iterator<std::string>(std::cout, "\n"));
42 }

```

lists/list0101.cpp

- (a) O que este programa faz? Neste primeiro momento, não será necessário compreender todos os detalhes do código, mas perceba como os comentários ajudam na compreensão do programa, assim como a indentação ajuda a perceber a estrutura do código-fonte. Perceba, também, como uma boa escolha dos nomes de variáveis melhora a legibilidade do código.
- (b) Observe a função `read()` (linhas 14-19). Qual a vantagem da utilização de subprogramas? Qual o princípio por trás desta estruturação do código?
- (c) O que a seguinte linha de compilação realiza?

```
g++ -o list0101 -pedantic -std=c++11 list0101.cpp
```

- (d) Execute o arquivo resultante da compilação. Como você pode testá-lo?

5. Considere o código mostrado na listagem `list0201.cpp`:

```
1  /** @file list0201.cpp */
2  /** Listing 2-1. Reading Test */
3  /// Read the program and determine what the program does.
4
5  #include <iostream>
6  #include <istream>
7  #include <limits>
8  #include <ostream>
9
10 int main()
11 {
12     int min(std::numeric_limits<int>::max());
13     int max(std::numeric_limits<int>::min());
14     bool any(false);
15     int x;
16     while (std::cin >> x)
17     {
18         any = true;
19         if (x < min)
20             min = x;
21         if (x > max)
22             max = x;
23     }
24
25     if (any)
26         std::cout << "min = " << min << "\nmax = " << max << '\n';
27 }
```

lists/list0201.cpp

- (a) O que este programa faz?
- (b) Nas linhas 1-3 podemos identificar duas diferentes maneiras de comentar o código. Defina as regras adotadas por tais formas de comentários. O que acontecerá se trocarmos o comentário `///` por outro no estilo `/**...*/` no código acima?
- (c) Qual a função das diretivas das linhas 5-8?
- (d) Justifique a inclusão das bibliotecas para o funcionamento do programa `list0201.cpp`. Para isso, relacione as bibliotecas incluídas com os comandos utilizados neste programa.
- (e) Na linha 7, substitua `limits` por `stimil`. O que acontecerá na compilação do programa?
- (f) Na linha 10, substitua `main` por `MAIN`. O que acontecerá na execução do programa?
- (g) Em quais condições o teste do `while` (linha 16) será avaliado para verdadeiro?
- (h) Suponha uma nova versão deste código com a linha 19 modificada para `if (x)`.
 - i. O que você espera acontecer ao compilar o programa?
 - ii. O que você espera acontecer quando executar o programa?
 - iii. O que você espera obter como resultado ao fornecer os valores 0, 1, 2 e 3 como entrada?
 - iv. O que você espera obter como resultado ao fornecer os valores 3, 2, 1, e 0 como entrada?
 - v. Explique os resultados obtidos por meio da execução do programa modificado.

6. Implemente e compile o programa abaixo.

```
/** @file list0203.cpp */
/** Listing 2-3. Determining the Number of Bits in a bool. */
#include <iostream>
#include <limits>
#include <ostream>

int main()
{
    // Note that "digits" means binary digits, i.e., bits.
    std::cout << "bits per bool: " << std::numeric_limits<bool>::digits << '\n';
    //std::cout << "not quoted; \"in quotes\", not quoted";
}
```

lists/list0203.cpp

- (a) Explique o resultado de sua execução.
- (b) Considerando as listagens `list0201.cpp` e `list0203.cpp`, o que você deduz sobre o comportamento do template `numeric_limits`?
- (c) Crie uma tabela resumindo os tipos básicos do C++.
- (d) Inclua o seguinte comando imediatamente após a linha 10.

```
std::cout << "not quoted; \"in quotes\", not quoted";
```

O resultado deste comando foi o esperado por você?

7. A listagem a seguir mostra um programa simples que executa entradas e saídas de dados:

```
/** @file list0501.cpp */
/** Listing 5-1. Demonstrating Input and Output */
#include <iostream>
#include <istream>
#include <ostream>

int main()
{
    std::cout << "Enter a number: ";
    int x;
    std::cin >> x;
    std::cout << "Enter another number: ";
    int y;
    std::cin >> y;

    int z(x + y);
    std::cout << "The sum of " << x << " and " << y << " is " << z << "\n";
}
```

lists/list0501.cpp

- (a) Suponha que usuário informa os números 42 e 21 como valores de entrada, separados por espaços. Qual a saída esperada? Execute o programa e verifique sua previsão.
- (b) Para cada valor de entrada mostrado a seguir, explique o resultado obtido:
 - i. 42×21

- ii. xyz
- iii. 42-21
- iv. 42.21

(c) Como a falta de inicialização das variáveis influencia nos resultados anteriores?

8. Suponha que você precisa escrever um programa que imprima os quadrados e cubos dos inteiros entre 1 e 20, de maneira a obter o seguinte resultado:

N	N ²	N ³
1	1	1
2	4	8
3	9	27
4	16	64
⋮	⋮	⋮
20	400	8000

Para ajudá-lo neste tarefa, tome o código esquemático a seguir como ponto de partida. Dica: observe os cabeçalhos incluídos neste esquema!

```

1  /** @file list0801.cpp */
2  /** Listing 8-1. Print a Table of Squares and Cubes */
3  #include <iomanip>
4  #include <iostream>
5  #include <ostream>
6
7  int main()
8  {
9      std::cout << " N    N^2    N^3\n";
10     for (int i = 1; i != 21; ++i)
11     {
12         // fill in the loop body here
13     }
14 }

```

lists/list0801.cpp

9. O seguinte programa poderá ser compilado com sucesso? Identifique todos os novos recursos introduzidos e justifique sua resposta.

```

1  /** @file list0803.cpp */
2  /** Listing 8-3. Using Alternative Fill Characters */
3  #include <iomanip>
4  #include <iostream>
5  #include <ostream>
6
7  int main()
8  {
9      using namespace std;
10
11     int day(14);
12     int month(3);
13     int year(2006);

```

```

int dollars(42);
int cents(7);

// Print date in USA order.
cout << "Date: " << setfill('0') << setw(2) << month
        << '/' << setw(2) << day
        << '/' << setw(2) << year << '\n';

cout << "Pay to the order of: CASH\n";
cout << "The amount of $" << setfill('*') << setw(8) << dollars << ' '
        << setfill('0') << setw(2) << cents << '\n';
}

```

lists/list0803.cpp

10. Considere o seguinte programa:

```

/** @file list0804.cpp */
/** Listing 8-4. Exploring Field Width, Padding, and Alignment */
#include <iomanip>
#include <ios>
#include <iostream>
#include <ostream>

int main()
{
    using namespace std;
    cout << '|' << setfill('*') << setw(6) << 1234 << '|' << '\n';
    cout << '|' << left << setw(6) << 1234 << '|' << '\n';
    cout << '|' << setw(6) << -1234 << '|' << '\n';
    cout << '|' << right << setw(6) << -1234 << '|' << '\n';
}

```

lists/list0804.cpp

- O que você espera como resultado da execução deste programa?
- Crie um programa que utilize larguras de campos, caracteres de preenchimento e manipuladores de alinhamento para produzir o seguinte resultado:

```

000042
420000
42
-42-

```

11. O seguinte programa imprime dez números não negativos:

```

/** @file list0701.cpp */
/** Listing 7-1. Using a for Loop to Print Ten Non-negative Numbers */
#include <iostream>
#include <ostream>

int main()
{
    for (int i(0); i != 10; i = i + 1)
        std::cout << i << '\n';
}

```

```
}
```

10

lists/list0701.cpp

Considerando este exemplo, crie um programa para computar e imprimir a soma dos inteiros de 10 a 20 a partir do esboço dado pela pelo programa `list0704.cpp`:

```
/** @file list0704.cpp */
```

1

```
/** Listing 7-4. Compute Sum of Integers from 10 to 20 */
```

2

```
#include <iostream>
```

3

```
#include <ostream>
```

4

```
int main()
```

5

```
{
```

6

```
    int sum(0);
```

7

```
    // Fill in the loop here.
```

8

```
    std::cout << "Sum of 10 to 20 = " << sum << '\n';
```

9

```
}
```

10

lists/list0704.cpp

12. Considere o programa definido pela listagem `list1902.cpp`:

```
/** @file list1902.cpp */
```

1

```
/** Listing 19-2. Separating Function Declarations from Definitions */
```

2

```
#include <iostream>
```

3

```
#include <istream>
```

4

```
#include <ostream>
```

5

```
#include <string>
```

6

```
void ignore_line();
```

7

```
int prompted_read(std::string prompt);
```

8

```
void print_result(int count, int sum);
```

9

```
/** Main program.
```

10

```
 * Read integers from the standard input and print statistics about them.
```

11

```
 */
```

12

```
int main()
```

13

```
{
```

14

```
    int sum(0);
```

15

```
    int count(0);
```

16

```
    while (std::cin)
```

17

```
    {
```

18

```
        int x(prompted_read("Value: "));
```

19

```
        if (std::cin)
```

20

```
        {
```

21

```
            sum = sum + x;
```

22

```
            ++count;
```

23

```
        }
```

24

```
    }
```

25

```
    print_result(count, sum);
```

26

```
}
```

27

```

31  /** Prompt the user, then read a number, and ignore the rest of the line.
32   * @param prompt the prompt string
33   * @return the input number or -1 for end-of-file
34   */
35
36  int prompted_read(std::string prompt)
37  {
38      std::cout << prompt;
39      int x(-1);
40      std::cin >> x;
41      ignore_line();
42      return x;
43  }
44
45  /** Ignore the rest of the input line. */
46  void ignore_line()
47  {
48      char c;
49      while (std::cin.get(c) and c != '\n')
50          /*empty*/;
51  }
52
53  /** Print the statistics.
54   * @param count the number of values
55   * @param sum the sum of the values
56   */
57  void print_result(int count, int sum)
58  {
59      if (count == 0)
60      {
61          std::cout << "no data\n";
62          return;
63      }
64
65      std::cout << "\ncount = " << count;
66      std::cout << "\nsum   = " << sum;
67      std::cout << "\nmean  = " << sum/count << '\n';
68  }

```

lists/list1902.cpp

- O que este programa faz?
- As linhas 8-10 apresentam declarações de funções. Estas declarações são obrigatórias?
- Qual a diferença entre a declaração e a definição de uma função?
- O programa apresentará resultados diferentes se a variável `x` for inicializada com 0 na linha 39? Por que?
- Utilize o GDB (GNU Debugger) para rastrear a execução do executável obtido pela compilação desta listagem. Durante a compilação devemos utilizar o parâmetro `-g` para que os símbolos necessários ao processo de depuração (como nomes das variáveis e referências às linhas do código fonte, por exemplo) sejam incluídos no executável gerado. Consulte as referências ao GDB indicadas no tópico de aula referente a este subproblema no SIGAA.