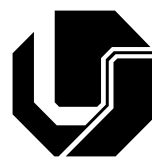




Universidade Federal de Uberlândia
Faculdade de Engenharia Elétrica
Experimental de Metrologia



Programação Procedimental

Lista 4 - Laboratório

Dr. Marcelo Barros de Almeida

Matheus Souza Da Costa

12021EEL021

Lista de ilustrações

Figura 1 – Questão 1	6
Figura 2 – Questão 2	7
Figura 3 – Questão 3	8
Figura 4 – Questão 4	9
Figura 5 – Questão 5	10
Figura 6 – Questão 6	11
Figura 7 – Questão 7	12
Figura 8 – Questão 8	13
Figura 9 – Questão 9 A	14
Figura 10 – Questão 9 B	15
Figura 11 – Questão 10	16

Sumário

1	Questão	4
2	Questão	4
3	Questão	4
4	Questão	4
5	Questão	4
6	Questão	4
7	Questão	4
8	Questão	4
9	Questão	4
10	Questão	5

1 Questão

Resposta conforme imagem 1

2 Questão

Resposta conforme imagem 2

3 Questão

Resposta conforme imagem 3

4 Questão

Resposta conforme imagem 4

5 Questão

Resposta conforme imagem 5

6 Questão

Resposta conforme imagem 6

7 Questão

Resposta conforme imagem 7

8 Questão


Resposta conforme imagem 8

9 Questão

Resposta conforme imagens 9 e 10

10 Questão

Resposta conforme imagem 11



```

1  /*
2   * Arquivo: questão1.c
3   * Compilação: gcc -std=c99 questão1.c -o questão1
4   * Autor: Matheus Souza Da Costa 12021EEL021
5   */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char str1[81], str2[81];
17
18     printf("Digite a primeira string (até 80 caracteres): ");
19     fgets(str1, sizeof(str1), stdin);
20
21     printf("Digite a segunda string (até 80 caracteres): ");
22     fgets(str2, sizeof(str2), stdin);
23
24     // Remova o caractere de nova linha no final das strings
25     str1[strcspn(str1, "\n")] = '\0';
26     str2[strcspn(str2, "\n")] = '\0';
27
28     int result = strcmp(str1, str2);
29
30     if (result == 0) {
31         printf("0\n");
32     } else if (result < 0) {
33         printf("-1\n");
34     } else {
35         printf("1\n");
36     }
37
38
39     return EXIT_SUCCESS;
40 }
41

```

Figura 1 – Questão 1

```

1  /*
2  * Arquivo: questao2.c
3  * Compilação: gcc -std=c99 questao2.c -o questao2
4  * Autor: Matheus Souza Da Costa 12021EEL021
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char input[81], output[81];
17
18     printf("Digite uma string (até 80 caracteres): ");
19     fgets(input, sizeof(input), stdin);
20
21     int input_len = strlen(input);
22     int output_len = 0;
23     int space_count = 0;
24
25     for (int i = 0; i < input_len; i++) {
26         if (input[i] != ' ') {
27             // Se o caractere não for um espaço, copie-o para a string de saída
28             output[output_len] = input[i];
29             output_len++;
30             space_count = 0; // Zera a contagem de espaços consecutivos
31         } else {
32             // Se for um espaço, conte-o apenas se não houver muitos espaços consecutivos
33             if (space_count < 1) {
34                 output[output_len] = input[i];
35                 output_len++;
36                 space_count++;
37             }
38         }
39     }
40
41     // Adicione o caractere nulo de terminação da string
42     output[output_len] = '\0';
43
44     printf("String sem espaços extras: %s\n", output);
45
46     return EXIT_SUCCESS;
47 }
48

```

Figura 2 – Questão 2

```

1  /*
2  * Arquivo: questao3.c
3  * Compilação: gcc -std=c99 questao3.c -o questao3
4  * Autor: Matheus Souza Da Costa 12021EEL021
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16
17     char str1[81], str2[81], str3[81];
18
19     printf("Digite a primeira string: ");
20     fgets(str1, sizeof(str1), stdin);
21
22     printf("Digite a segunda string: ");
23     fgets(str2, sizeof(str2), stdin);
24
25     printf("Digite a terceira string: ");
26     fgets(str3, sizeof(str3), stdin);
27
28     // Remova a quebra de linha ('\n') das strings lidas
29     str1[strcspn(str1, "\n")] = '\0';
30     str2[strcspn(str2, "\n")] = '\0';
31     str3[strcspn(str3, "\n")] = '\0';
32
33     // Encontre a string de menor valor (ordem alfabética) usando strcmp
34     char *min_str = str1;
35     if (strcmp(str2, min_str) < 0) {
36         min_str = str2;
37     }
38     if (strcmp(str3, min_str) < 0) {
39         min_str = str3;
40     }
41
42     // Concatene as strings em ordem alfabética
43     char result[243]; // Tamanho total máximo de 3 strings de 80 caracteres
44     strcpy(result, min_str);
45
46     if (min_str == str1) {
47         strcat(result, str2);
48         strcat(result, str3);
49     } else if (min_str == str2) {
50         strcat(result, str1);
51         strcat(result, str3);
52     } else {
53         strcat(result, str1);
54         strcat(result, str2);
55     }
56
57     printf("Strings concatenadas em ordem alfabética: %s\n", result);
58
59     return EXIT_SUCCESS;
60 }
61

```

Figura 3 – Questão 3


```
1  /*
2   * Arquivo: questao4.c
3   * Compilação: gcc -std=c99 questao4.c -o questao4
4   * Autor: Matheus Souza Da Costa 12021EEL021
5   */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <ctype.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char str[81];
17
18     printf("Digite uma string (até 80 caracteres): ");
19     fgets(str, sizeof(str), stdin);
20
21     // Loop para percorrer cada caractere da string
22     for (int i = 0; str[i] != '\0'; i++) {
23         // Verifica se o caractere é uma letra maiúscula
24         if (isupper(str[i])) {
25             // Converte a letra maiúscula em minúscula
26             str[i] = tolower(str[i]);
27         }
28     }
29
30     printf("String convertida para minúsculas: %s\n", str);
31
32     return EXIT_SUCCESS;
33 }
34
```

Figura 4 – Questão 4

```

1  /*
2   * Arquivo: questao5.c
3   * Compilação: gcc -std=c99 questao5.c -o questao5
4   * Autor: Matheus Souza Da Costa 12021EEL021
5   */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16
17     char str1[100], str2[100];
18     int count1[256] = {0}; // Array para contar os caracteres da primeira string
19     int count2[256] = {0}; // Array para contar os caracteres da segunda string
20     int len1, len2;
21
22     printf("Digite a primeira string: ");
23     gets(str1); // Leitura da primeira string
24
25     printf("Digite a segunda string: ");
26     gets(str2); // Leitura da segunda string
27
28     // Calcula o comprimento das strings
29     len1 = strlen(str1);
30     len2 = strlen(str2);
31
32     // Se as strings tiverem comprimentos diferentes, elas não podem ser anagramas
33     if (len1 != len2) {
34         printf("Nao sao anagramas.\n");
35         return 0;
36     }
37
38     // Conta os caracteres da primeira string
39     for (int i = 0; i < len1; i++) {
40         count1[(int)str1[i]]++;
41     }
42
43     // Conta os caracteres da segunda string
44     for (int i = 0; i < len2; i++) {
45         count2[(int)str2[i]]++;
46     }
47
48     // Compara as contagens de caracteres
49     for (int i = 0; i < 256; i++) {
50         if (count1[i] != count2[i]) {
51             printf("Nao sao anagramas.\n");
52             return 0;
53         }
54     }
55
56     printf("Sao anagramas.\n");
57
58
59     return EXIT_SUCCESS;
60 }
61

```

Figura 5 – Questão 5

```

1  /*
2  * Arquivo: questao6.c
3  * Compilação: gcc -std=c99 questao6.c -o questao6
4  * Autor: Matheus Souza Da Costa 12021EEL021
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char str1[100], str2[100];
17
18     printf("Digite a primeira string: ");
19     gets(str1); // Leitura da primeira string
20
21     printf("Digite a segunda string: ");
22     gets(str2); // Leitura da segunda string
23
24     int len1 = strlen(str1);
25     int len2 = strlen(str2);
26
27     // Cria um array para marcar quais caracteres da primeira string estão presentes
28     int charSet[256] = {0};
29
30     // Marca os caracteres da primeira string no array charSet
31     for (int i = 0; i < len1; i++) {
32         charSet[(int)str1[i]] = 1;
33     }
34
35     // Cria um índice para a string resultante
36     int resultIndex = 0;
37
38     // Remove os caracteres da primeira string da segunda string
39     for (int i = 0; i < len2; i++) {
40         if (charSet[(int)str2[i]] == 0) {
41             // Se o caractere não estiver na primeira string, copia-o para a string resultante
42             str2[resultIndex] = str2[i];
43             resultIndex++;
44         }
45     }
46
47     // Adiciona o caractere nulo de término de string
48     str2[resultIndex] = '\0';
49
50     printf("String resultante: %s\n", str2);
51
52     return EXIT_SUCCESS;
53 }
54
55

```

Figura 6 – Questão 6

```

1  /*
2  * Arquivo: questao7.c
3  * Compilação: gcc -std=c99 questao7.c -o questao7
4  * Autor: Matheus Souza Da Costa 12021EEL021
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char T[1000], p[100];
17
18     printf("Digite a string T: ");
19     gets(T); // Leitura da string T
20
21     printf("Digite a string p: ");
22     gets(p); // Leitura da string p
23
24     int lenT = strlen(T);
25     int lenP = strlen(p);
26
27     if (lenT < lenP) {
28         printf("A string p nao pode ser maior do que a string T.\n");
29         return 1;
30     }
31
32     int i, j, found;
33
34     for (i = 0; i <= lenT - lenP; i++) {
35         found = 1;
36
37         for (j = 0; j < lenP; j++) {
38             if (T[i + j] != p[j]) {
39                 found = 0;
40                 break;
41             }
42         }
43
44         if (found) {
45             printf("%d ", i);
46         }
47     }
48
49     printf("\n");
50
51     return EXIT_SUCCESS;
52 }

```

Figura 7 – Questão 7

```

1  /*
2  * Arquivo: questao8.c
3  * Compilação: gcc -std=c99 questao8.c -o questao8
4  * Autor: AUTHOR
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10 #include <string.h>
11
12
13
14
15 int isSubsequence(const char *T1, const char *T2) {
16     int lenT1 = strlen(T1);
17     int lenT2 = strlen(T2);
18
19     int i = 0; // Índice para T1
20     int j = 0; // Índice para T2
21
22     while (i < lenT1 && j < lenT2) {
23         if (T1[i] == T2[j]) {
24             i++; // Avança o índice de T1
25         }
26         j++; // Avança o índice de T2
27     }
28
29     return (i == lenT1); // Se i for igual ao comprimento de T1, T1 é subsequência de T2
30 }
31
32 int main(void) {
33     #if defined(_WIN32) || defined(_WIN64)
34         system("chcp 65001"); // troca para UTF-8
35     #endif
36
37     char T1[100], T2[100];
38
39     printf("Digite a string T1: ");
40     gets(T1); // Leitura da string T1
41
42     printf("Digite a string T2: ");
43     gets(T2); // Leitura da string T2
44
45     if (isSubsequence(T1, T2)) {
46         printf("T1 e uma subsequencia de T2.\n");
47     } else {
48         printf("T1 nao e uma subsequencia de T2.\n");
49     }
50
51     return EXIT_SUCCESS;
52 }
53

```

Figura 8 – Questão 8

```

1  /*
2   * Arquivo: questao9.c
3   * Compilação: gcc -std=c99 questao9.c -o questao9
4   * Autor: Matheus Souza Da Costa 12021EEL021
5   */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main(void) {
12     #if defined(_WIN32) || defined(_WIN64)
13         system("chcp 65001"); // troca para UTF-8
14     #endif
15
16     char original[81], inversa[81];
17
18     // Solicita ao usuário para inserir uma string de até 80 caracteres
19     printf("Digite uma string (até 80 caracteres): ");
20     fgets(original, sizeof(original), stdin);
21
22     // Remove a quebra de linha (\n) no final da string, se existir
23     original[strcspn(original, "\n")] = '\0';
24
25     int tamanho = strlen(original);
26
27     // Inverte a string original e a armazena na string inversa
28     for (int i = 0; i < tamanho; i++) {
29         inversa[i] = original[tamanho - i - 1];
30     }
31     inversa[tamanho] = '\0';
32
33     // Exibe a string inversa
34     printf("A string inversa é: %s\n", inversa);
35
36     return EXIT_SUCCESS;
37 }
38
39 }
40

```

Figura 9 – Questão 9 A

```

1  /*
2   * Arquivo: questao9.c
3   * Compilação: gcc -std=c99 questao9.c -o questao9
4   * Autor: Matheus Souza Da Costa 12021EEL021
5   */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10
11 int main() {
12     char str[81];
13
14     // Solicita ao usuário para inserir uma string de até 80 caracteres
15     printf("Digite uma string (até 80 caracteres): ");
16     fgets(str, sizeof(str), stdin);
17
18     // Remove a quebra de linha (\n) no final da string, se existir
19     str[strcspn(str, "\n")] = '\0';
20
21     int tamanho = strlen(str);
22
23     // Inverte a string diretamente no vetor original
24     for (int i = 0; i < tamanho / 2; i++) {
25         char temp = str[i];
26         str[i] = str[tamanho - i - 1];
27         str[tamanho - i - 1] = temp;
28     }
29
30     // Exibe a string invertida no vetor original
31     printf("A string inversa é: %s\n", str);
32
33     return 0;
34 }
35

```

Figura 10 – Questão 9 B

```

1  /*
2  * Arquivo: questao10.c
3  * Compilação: gcc -std=c99 questao10.c -o questao10
4  * Autor: Matheus Souza Da Costa 12021EEL021
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9  #include <string.h>
10 #include <ctype.h>
11
12 int main(void) {
13     #if defined(_WIN32) || defined(_WIN64)
14         system("chcp 65001"); // troca para UTF-8
15     #endif
16
17     char str[81];
18
19     // Solicita ao usuário para inserir uma string de até 80 caracteres
20     printf("Digite uma string (até 80 caracteres): ");
21     fgets(str, sizeof(str), stdin);
22
23     // Remove a quebra de linha (\n) no final da string, se existir
24     str[strcspn(str, "\n")] = '\0';
25
26     int tamanho = strlen(str);
27
28     // Remove espaços em branco e converte todos os caracteres para minúsculas
29     char novaString[81];
30     int j = 0;
31     for (int i = 0; i < tamanho; i++) {
32         if (!isspace(str[i])) {
33             novaString[j] = tolower(str[i]);
34             j++;
35         }
36     }
37     novaString[j] = '\0';
38
39     // Verifica se a string é um palíndromo
40     int isPalindromo = 1;
41     int len = strlen(novaString);
42     for (int i = 0; i < len / 2; i++) {
43         if (novaString[i] != novaString[len - i - 1]) {
44             isPalindromo = 0;
45             break;
46         }
47     }
48
49     // Exibe o resultado
50     if (isPalindromo) {
51         printf("A string é um palíndromo.\n");
52     } else {
53         printf("A string não é um palíndromo.\n");
54     }
55
56     return EXIT_SUCCESS;
57 }
58
59 }
60

```

Figura 11 – Questão 10