

Nome: Júlia Branonese Folger

Data: 04/07/2024

0,2

1. (2,0) Analise os códigos fornecidos e identifique possíveis problemas, erros ou oportunidades de melhoria. Faça as correções necessárias e explique as razões das mudanças realizadas.

Código 1:

```

1 #include<stdio.h>
2 #include<string.h>
3
4 int saoIguais(char str1[], char str2[]) {
5     if (strlen(str1) != strlen(str2)) {
6         return 0;
7     }
8     for (int i = 0; i <= strlen(str1); i++) {
9         if (str1[i] != str2[i]) {
10             return 0;
11         }
12     }
13     return 1;
14 }
15
16 void inverterString(char str[]) {
17     int inicio = 0;
18     int fim = strlen(str) - 1;
19     while (inicio < fim) {
20         char temp = str[inicio];
21         str[inicio] = str[fim];
22         str[fim] = temp;
23         inicio++; i, 2
24         fim--; 2, 1
25     }
26 }
27
28 int main() {
29     char palavra1[100], palavra2[100];
30     printf("Digite a primeira palavra: ");
31     scanf("%s", palavra1);
32     printf("Digite a segunda palavra: ");
33     scanf("%s", palavra2);
34     if (saoIguais(palavra1, palavra2) == 0) {
35         printf("são iguais.\n");
36     }
37     else {
38         printf("são diferentes.\n");
39     }
40     inverterString(palavra1);
41     printf("%s\n", palavra1);
42     return 0;
43 }
```

Problemas e/ou erros:

- Na função "saoIguais" ele está retornando 0 se as palavras forem diferentes, porém no if da linha 34, diz que se o retorno for igual a 0, as palavras são iguais.
- Nos scanf() das linhas 31 e 33, falta %s antes da palavra e palavras.
- Para passar o dado de uma string para outra é preciso usar o strcpy().

Correções e/ou melhorias: (linhas 21 e 22)

- Por se tratar de strings, poderia receber-las por gets() ou fgets() nas linhas 31 e 33.

- Na linha 34, ao invés de criar uma função específica, poderia ser usada a função strcmp (palavra1, palavra2).

- Na linha 41, poderia ser usado puts() os invés de printf().

0,8

```
Código 2:
1 #include <stdio.h>
2 #include <string.h>
3
4 void concatenarStrings(char str1[], char str2[]) {
5     int tamanho1 = strlen(str1);
6     int tamanho2 = strlen(str2);
7     for (int i = 0; i <= tamanho2; i++) {
8         str1[tamanho1 + i] = str2[i];
9     }
10 }
11
12 int main() {
13     int N = 20;
14     char string1[N] = "Hello\0";
15     char string2[N] = "world!\0";
16     concatenarStrings(string1, string2);
17     printf("Strings concatenadas: %s\n", string1);
18     return 0;
19 }
```

Problemas e/ou erros:

- As palavras estavam escritas ~~co-~~
juntas. *

- Por passar o endereço de uma string
para outro é necessário usar o strcpy(). X
↳ Linha 8.

Correções e/ou melhorias:

- Ao invés de criar uma função
para concatenar, poderia ser usado
a função strcat().

* Para resolver esse problema, poderia colocar
antes da linha 7: str1[tamanho1+1] = '\0'
e na linha 8 substituir tamanho1+i por
tamanho1+i+1.

1, D

2. (1,0) Analise o código abaixo e marque V para as afirmações verdadeiras e F para as falsas. Justifique as falsas.

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main() {
5     char str1[20] = "Programming";
6     char str2[20] = "is";
7     char str3[20] = "fun";
8
9     strcat(str1, str2); → Programmingis
10    strcat(str3, str2); → funis
11
12    printf("str1: %s\n", str1);
13    printf("str3: %s\n", str3);
14
15    return 0;
16 }
```

1-(V) O resultado da execução do código será "Programmingis" para str1.

2-(V) O resultado da execução do código será "funis" para str3.

3-(V) A função strcat(str1, str2); garante que str1 será corretamente terminada com o caractere nulo ('\0') após a concatenação.

4-(F) A função strcat(str3, str2); garante que str3 terá um total de 4 caracteres após a concatenação.

5-(F) O código não irá compilar corretamente, pois há erros de sintaxe.

4→ Como a função strcat() está concatenando duas palavras, uma com 3 caracteres, outra com 2, no final da str3, terá 5 caracteres (3+2).

5→ O código compilará corretamente..

3. Observe o algoritmo abaixo. Analise os itens a seguir e assinale a(s) opção(ões) CORRETA(S), justificando sua(s) resposta(s):

```

1 #include<stdio.h>
2 #include<string.h>
3
4 void func(char p[100], int v, char s[100]) {
5     int i,k,j;
6     for(i=0, j=0; i<=v; i++){
7         if((p[i]==' ')||(p[i]=='\0')){
8             for(k=0; k<i; k++){
9                 s[j+k]=p[i-1-k];
10            }
11            s[j+k]=' ';
12            j=i+1;
13        }
14    }
15    s[v]='\0';
16 }
17
18 int main(){
19     int t=0,i,j,k,c=1;
20     char f[100], fi[100], fg[4>{"fim"};
21     while(c == 1){
22         printf("Entrada: ");
23         gets(f); f = cosa
24         t = strlen(f); b = t
25         if((strcmp(strlwr(f),fg))!=0){
26             func(f,t,fi);
27             printf("%s\n",fi);
28         }
29         else{
30             c=0;
31         }
32     }
33     return 0;
34 }
35

```

$$\begin{array}{ll}
 0+0 & 5-1-0=4 = a \\
 0+1 & 5-1-1=3 = a \\
 0+2 & 5-1-2=2 = a \\
 0+3 & 5-1-3=1 = \text{ } \\
 0+4 & 5-1-4=0 = 10
 \end{array}$$

- a. (E) Objetivo deste algoritmo é identificar quando "fim" é digitado.
- b. (E) Se a entrada for "teste 123" na linha 23, o valor mostrado na linha 25 será "321 etset".
- c. (C) A função 'func' inverte as palavras de 'p' e armazena em 's'.
- d. (C) O laço de repetição da linha 21, é encerrado somente quando "fim" for o valor de entrada na linha 23.
- e. (E) A função 'func', na linha 26, poderia ser substituída por strcpy(f,fi).

a) O objetivo é inverter as palavras, mesmo se for uma frase, a sequência da mesma estará correta, só as ~~as~~ palavras estarão no contrário, isso se a palavra de entrada for diferente de "fim".
 Ex: A cosa vermelha -> A cosa alemanha.
 entrada -> palavras na ordem inversa -> saída

- b) A saída correta será "etset 321", pois a função inverte as palavras por paços (até o espaço).
 Conforme vai invertendo, (~~o~~saída) é formada a string "\n" na linha 9.
- c) A palavra fim é o ponto de parada, independente se for uma letra maiúscula ou não, por causa do strlwr da linha 25. Se a palavra for "fim", c sera igual 0.-> linha 30 (else)
- d) Não, a função inverte as palavras, coisa que a função strcpy() não faz.

4. (2,0) Você foi contratado para desenvolver um sistema de gerenciamento de estoque para uma loja. O sistema deve utilizar ~~structs~~ para representar cada produto, com os seguintes dados: código do produto, nome do produto, preço unitário e quantidade em estoque. Implemente as seguintes funcionalidades:

- 1, 8
- **Remoção de Produtos:** Uma função que permite ao usuário remover um produto do estoque, informando o código do produto.
 - **Valor Total do Estoque:** Uma função que calcula e imprime o valor total do estoque, considerando o preço unitário e a quantidade de cada produto.

Dica: Assuma que a inserção dos N produtos já esteja implementada e funciona sem problemas. Crie apenas o registro e as funções solicitadas.

```
struct loja {
    int codigo;
    char nome[25];
    float preco-unitario;
    int qtd-estoque;
}
```

```
(void) remove-produto (struct loja produto[N]) {
```

```
int remove;
printf("Código de remoção");
scanf("%d", &remove);
```

```
for (int i=0; i<N; i++) {
    if (produto[i].codigo == remove) {
```

```
        for (int j=i; j<N-1; j++) {
            produto[j].codigo = produto[j+1].codigo;
            strcpy(produto[j].nome, produto[j+1].nome);
            produto[j].preco-unitario = produto[j+1].preco-unitario;
            produto[j].qtd-estoque = produto[j+1].qtd-estoque;
```

```
}
```

```
produto[N].codigo = -1;
strcpy(produto[N].nome, ' ');
produto[N].preco-unitario = 0;
produto[N].qtd-estoque = 0;
```

```
RETUR N - 1;
```

faltou decrementar N

```
(void) valor-total (struct loja produto[N]) {
```

```
float valor-total = 0, aux;
```

```
for (int i=0; i<N; i++) {
```

```
    aux = produto[i].preco-unitario * produto[i].qtd-estoque;
    valor-total += aux;
```

```
printf("Valor total do estoque: R$%.2f", valor-total);
```

```
}
```