Linguagem C para Engenharia

Lista de Problemas

(ENG 03049)

Bardo E.J. Bodmann

Versão 4.3 (14 de maio de 2012)

Escola de Engenharia – EE
Universidade Federal do Rio Grande do Sul – UFRGS

1 Lista de Problemas

1. O que faz o seguinte programa?

```
#include <stdio.h>

int main()
{
int x;

scanf("%d",&x);
printf("%d",x);

return (0);
}
```

2. Escreva uma função que some dois inteiros e retorne o valor da soma.

3. (a) Escreva um programa que leia um caracter digitado pelo usuário, imprima o caracter digitado e o código ASCII correspondente a este caracter.
   (b) Escreva um programa que leia duas strings e as coloque na tela. Imprima também a segunda letra de cada string.

4. (a) Explique porque está errado fazer $if\ (num = 10)$. O que irá acontecer?
   (b) Escreva um programa que coloque os números de 1 a 100 na tela na ordem inversa (começando em 100 e terminando em 1).
   (c) Escreva um programa que leia uma string, conte quantos caracteres desta string são iguais a 'a' e substitua os que forem iguais a 'a' por 'b'. O programa deve imprimir o número de caracteres modificados e a string modificada.

5. Escreva um programa que declare uma variável inteira global e atribua o valor 10 a ela. Declare outras 5 variáveis inteiras locais ao programa principal e atribua os valores $20, 30, ..., 60$ a elas. Declare 6 variáveis caracteres e atribua a elas as letras c, o, e, l, h, a. Finalmente, o programa deverá imprimir, usando todas as variáveis declaradas: As variáveis inteiras contem os números: $10, 20, 30, 40, 50, 60$. O animal contido nas variáveis caracteres e a coelha.

6. Diga o resultado das variáveis x, y e z depois da seguinte seqência de operações:

```
int x,y,z;
x=y=10;
z=++x;
x=-x;
y++;
x=x+y-(z--);
```

7. Diga se as seguintes expressões serão verdadeiras ou falsas:

```
((10>5)||(5>10))
(!(5==6)&&(5!=6)&&((2>1)||(5<=4)))
```

8. Faça um programa que apresente na tela a tabela de onversão de graus Celsius para Fahrenheit, de $-100C$ a $100C$. Use um incremento de $10C$. Utilize a formula de conversão $T_{Farenheit} = \frac{9F}{5^oC} * T_{Celsius} + 32F$.

9. Faça um programa em C para ler números em ponto flutuante (representando ângulos em radianos) e imprima o seno do numero. Ao ler o número zero, o programa devera terminar. Teste seu programa para verificar seu funcionamento. Para calcular o seno, você pode usar a função $sin(x)$, onde x e dado em radianos. Para usar esta função, você deve incluir o arquivo cabecalho *math.h*. Além disto, no instante de link você deve usar a biblioteca matemática. No *gcc*, usar a opcao *-lm* (Exemplo: *gcc programa.c -lm*).

10. Escreva um programa onde ao entrar com o número correspondente a um dia da semana seja escrito na tela o nome do dia.

11. Faça um programa que inverta uma string: leia a string com *gets* e armazene-a invertida em outra string. Use o comando *for* para ler a string até o seu final.

12. Refaça o programa anterior. Use o comando *while* para fechar o laço.

13. Escreva um programa que peça ao usuário que digite três números inteiros, correspondentes a dia , mês e ano. Teste os números recebidos, e em caso de haver algum inválido, repita a leitura até conseguir valores que estejam na faixa correta (dias entre 1 e 31, mês entre 1 e 12 e ano entre 1900 e 2100). Verifique se o mês e o número de dias batem (incluindo verificação de anos bissextos). Se estiver tudo certo imprima o número que aquele dia corresponde no ano. Note, que um ano é bissexto se for divisível por 4 e não for divisível por 100, exceto para os anos divisíveis por 400, que também são bissextos.

14. Faça um programa de conversão de base numérica. O programa deverá apresentar uma tela de entrada com as seguintes opções:

```
< Conversao de base >

1: decimal para hexadecimal

2: hexadecimal para decimal

3: decimal para octal

4: octal para decimal

5: Encerra

Informe sua opcao:
```

A partir da opção escolhida, o programa deverá pedir o número na base escolhida, lê-lo e apresentá-lo na base desejada. Em seguida, o programa deve perguntar ao usuário se ele deseja retornar ao menu principal ou finalizar o programa. Observe cada numero x pode ser representado em relação a uma base b arbitrária de forma generica:

$$x = m_0 b^0 + m_1 b^1 + \cdots + m_n b^n = \sum_{i=0}^{n} m_i b^i$$

15. Faça um programa que leia quatro palavras pelo teclado, e armazene cada palavra em uma string. Depois, concatene todas as strings lidas numa única string. Por fim apresente esta como resultado ao final do programa.

16. O que imprime o programa a seguir?

```
# include <stdio.h>

int main() {
int t, i, M[3][4];

for (t=0; t<3; ++t)
for (i=0; i<4; ++i)
M[t][i] = (t*4)+i+1;

for (t=0; t<3; ++t) {
for (i=0; i<4; ++i)
```

```
printf("%3d ", M[t][i]);

printf("\n");
}

return 0;
}
```

17. O que o programa a seguir faz? Qual será o resultado obtido se a string fornecida for a) "Ah! Eu to maluco!"b) "$5 * 4 + (3^3) + 4 * 5$"

```
#include <stdio.h>
#include <string.h>
#define TAM 20

int main() {
char s[TAM];
int c, i, j;

for (i=0, j=strlen(s)-1; i<j; i++, j--) {
c = s[i];
s[i] = s[j];
s[j] = c;
}

return 0;
}
```

18. a) Sendo p declarado de forma $int * p$;. Explique a diferença entre $p++$; $(*p)++$; $*(p++)$; O que quer dizer $*(p + 10)$;? Explique o que você entendeu da comparação entre ponteiros. b) Qual o valor de y no final do programa? Tente primeiro descobrir e depois verifique no computador o resultado. A seguir, escreva um /* comentário */ em cada comando de atribuição explicando o que ele faz e o valor da variável à esquerda do '=' após sua execução.

```
#include <stdio.h>

int main() {
int y, *p, x; y = 0;

p = &y;
x = *p;
x = 4;
(*p)++;
x--;
(*p) += x;

printf("y = %d\n", y);

return(0);
}
```

19. Fizemos a função *StrCpy()*. Faça uma função *StrLen()* e *StrCat()* que funcionem como as funções *strlen()* e *strcat()* de *string.h*, respectivamente.

20. Escreva a função *int strend(char *s, char *t)* que retorna 1 (um) se a cadeia de caracteres 't' ocorrer no final da cadeia 's', e 0 (zero) caso contrário.

21. Verifique o programa abaixo. Encontre o seu erro e corrija-o para que escreva o numero 10 na tela.

```
#include <stdio.h>

int main() {
int x, *p, **q;

p = &x;
q = &p;
x = 10;

printf("\n%d\n", &q);

return(0);
}
```

22. Escreva um programa que declare uma matriz 100×100 de inteiros. Você deve inicializar a matriz com zeros usando ponteiros. Preencha depois a matriz com os números de 1 a 10000 usando ponteiros.

23. Seja $a[]$ um vetor qualquer, independente de tipo e tamanho, e pa um ponteiro para o mesmo tipo de $a[]$. Responda V (verdadeiro) ou F (falso), justificando:

    - Após a atribuição $pa = \&a[0]$; pa e a possuem valores idênticos, isto é, apontam para o mesmo endereço;
    - A atribuição $pa = \&a[0]$; pode ser escrita como $pa = a$;
    - $a[i]$ pode ser escrito como $*(a + i)$;
    - $\&a[i]$ e $a + i$ são idênticos;
    - $a + i$ é o endereço do i-ésimo elemento após a;
    - $pa[i]$ é idêntico a $*(pa + i)$;
    - $pa = a$ é uma operação valida;
    - $pa++$ é uma operação valida;
    - $a = pa$ é uma operação valida $\Longrightarrow$ Um vetor não pode ter seu endereço modificado;
    - $a++$ é uma operação valida $\Longrightarrow$ Um vetor não pode ter seu endereço modificado.

24. O que está errado com os programas abaixo? Descubra e indique a solução para consertá-los. Execute-o no computador para ver se o erro foi resolvido. a)

```
void main() /* esse programa esta errado */
{
int x, *p;
x = 10;
*p = x;
}
```

b)

```
void main() /* esse programa esta errado */
{
int x, *p;
x = 10;
p = x;
printf("%d", *p);
}
```

25. Escreva a função *EDivisivel(int a, int b)*. A função deverá retornar 1 se o resto da divisão de a por b for zero. Caso contrário, a função deverá retornar zero.

26. Escreva um programa que faça uso da função *EDivisivel(int a, int b)*. Organize o seu programa em três arquivos: o arquivo *prog.c*, conterá o programa principal; o arquivo *func.c* conterá a função; o arquivo *func.h* conterá o protótipo da função. Compile os arquivos e gere o executável a partir deles.

27. Estude o seguinte programa e aponte o valor de cada variável sempre que solicitado:

```
#include <stdio.h>

int num;

int func(int a, int b)
{
a = (a+b)/2; /* Qual e o valor de a apos a atribuicao? */
num -= a;
return a;
}

main()
{
int first = 0, sec = 50;

num = 10;
num += func(first, sec);
/* Qual e o valor de num, first e sec */
/* antes e depois da atribuicao?     */

printf("\n\nConfira! num = %d\tfirst = %d\tsec = %d",num, first, sec);
```

28. Escreva uma função que receba duas variáveis inteiras e "zere"o valor das variáveis.

29. Escreva um programa que leia um vetor de inteiros pelo teclado e o apresente na tela. Crie uma função(*void levetor(int *vet, int dimensao)*) para fazer a leitura do vetor.

30. Escreva uma macro que retorne 1 se o seu argumento for um número ímpar e 0 se for um número par.

31. Escreva uma função que receba duas strings como argumentos e troque o conteúdo da primeira string com o conteúdo da segunda.

32. Escreva um programa que utilize os argumentos *argv* e *argc*. O programa deverá receber dois números e apresentar a soma dos dois. Veja que para isto você deverá ter também uma função que transforme uma string em um inteiro, pois o tipo de *argv* é *char*; logo você irá receber strings e deverá transformá-las em inteiros antes de somá-las.

33. Um problema tradicional é o de encontrar o enésimo termo da série de Fibonacci. As series de Fibonacci são de grande importância matemática, e a lei básica é que a partir do terceiro termo, todos os termos são a soma dos dois últimos. Os primeiros termos da seqência são: $1, 1, 2, 3, 5, 8, 13, 21, \ldots$. Escreva um programa que gere a série de Fibonacci.

34. O primeiro e o segundo termos são 1. O terceiro termo é 2 $(1 + 1)$. O quarto termo é 3 $(1 + 2)$. O quinto termo é 5 $(2 + 3)$. Faça uma função que encontre o n-ésimo termo da seqência de Fibonacci. Use recursividade.

35. Verifique o programa abaixo e tente identificar problemas que irão ocorrer em tempo de execução. (Você pode usar um debugger, ou colocar *printf*s para tentar 'ver' os valores das variáveis).

```
#define max(A,B) ((A>B) ? (A):(B))
#define min(A,B) ((A<B) ? (A):(B))
#define sqr(x) x*x

int main() {
int a = 10 ,b = 50, minimo, maximo, quad;
int z = 5;

minimo = min(a,b);
maximo = max(a++,b++);
quad = sqr(z+1);

return 0;
}
```

Verifique o que acontece também, se mudarmos a linha de declarações do programa *int a = 10 ,b = 50, minimo, maximo, quad*; por *float a = 10 ,b = 50, minimo, maximo, quad*.

36. Escreva um programa que leia nomes pelo teclado e os imprima na tela. Use as funções *puts* e *gets* para a leitura e impressão na tela.

37. Escreva um programa que leia (via teclado) e apresente na tela uma matriz 3×3 de inteiros. Utilize os novos códigos de formato aprendidos para que a matriz se apresente corretamente identada. Altere os tipos de dados da matriz *(int, float, double)* e verifique a formatação correta para a identação. Verifique também a leitura e impressão de números hexadecimais.

38. Escreva um programa que abra um arquivo texto e conte o número de caracteres presentes nele. Imprima o número de caracteres na tela.

39. Escreva um programa que leia uma lista de nomes e idades de um arquivo texto. Prepare um arquivo para ser lido com nomes e idades. Apresente os dados lidos em forma de tabela na tela. Use as funções de sua preferência, mas faça pelo menos duas versões do programa usando funções de leitura diferentes.

40. Escreva um programa que leia um arquivo do tipo base de dados em *.txt*, retire e apresente informações importantes na tela. Você pode criar o arquivo utilizando qualquer editor que permite gravar um arquivo no formato ASCII (Exemplo: Notepad do MSWindows, gedit ou kwrite do Linux, vi do Unix). O arquivo pode ser do tipo:

```
Nome:  Joao da Silva                                    Telefone:  +55 51 1234-5678
Endereco:  Rua do Joao, 168 $
Nome:  Maria da Silva                                   Telefone:  +55 51 8765-4321
Endereco:  Rua da Vila, 093 $
....
```

O símbolo $ (dólar) entre os dados é só para informar que acabou uma sequência de dados. Você pode usar um flag para encerrar, mas e melhor que use *feof()* (é mais elegante).

41. Faça um programa que multiplique duas matrizes. O programa deverá estar estruturado de maneira que: a) o usuário forneça as dimensões das matrizes (teste se as dimensões são compatíveis, isto é, se as matrizes podem ser multiplicadas); b) as matrizes sejam alocadas dinamicamente (você pode usar a função vista nesta pagina para isto); c) as matrizes sejam lidas pelo teclado (faca uma função para leitura das matrizes); d) as matrizes sejam, então, multiplicadas (faça uma função para a multiplicação); e) a matriz resultante seja apresentada em tela (faca uma função para apresentar a matriz na tela). Observações: a) Faça, também, alocação dinâmica da matriz resultante. b) O procedimento para a multiplicação de matrizes é o seguinte:

Suponha as matrizes $A(m \times n)$ e $B(n \times t)$

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1t} \\ b_{21} & b_{22} & \cdots & b_{2t} \\ \vdots & & & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nt} \end{pmatrix}$$

O elemento ij da matriz C é resultante da multiplicação da linha i de A pela coluna j de B. Portanto, a matriz $C(m \times t) = A * B$ será da seguinte forma:

$$C = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} + \cdots + a_{1n}b_{n1} & \cdots & a_{11}b_{1t} + a_{12}b_{2t} + \cdots + a_{1n}b_{nt} \\ \vdots & & \vdots \\ a_{m1}b_{11} + a_{m2}b_{21} + \cdots + a_{mn}b_{n1} & \cdots & a_{m1}b_{1t} + a_{m2}b_{2t} + \cdots + a_{mn}b_{nt} \end{pmatrix}$$

42. Escreva um programa fazendo o uso de *struct*'s. Você deverá criar uma *struct* chamada Ponto, contendo apenas a posição x e y (inteiros) do ponto. Declare 2 pontos, leia a posição (coordenadas x e y) de cada um e calcule a distância entre eles. Apresente no final a distância entre os dois pontos.

43. Seja a seguinte *struct* que é utilizada para descrever os produtos que estão no estoque de uma loja :

```
struct Produto {
char nome[30]; /* Nome do produto */
int codigo; /* Codigo do produto */
double preco; /* Preco do produto */
};
```

a) Escreva uma instrução que declare uma matriz de *Produto* com 10 itens de produtos; b) Atribua os valores "Pe de Moleque", 13205 e $R\$0,20$ aos membros da posição 0 e os valores "Cocada Baiana", 15202 e $R\$0,50$ aos membros da posição 1 da matriz anterior; c) Faça as mudanças que forem necessárias para usar um ponteiro para *Produto* ao invés de uma matriz de *Produtos*. Faça a alocação de memória de forma que se possa armazenar 10 produtos na área de memória apontada por este ponteiro e refaça as atribuições da letra b; d) Escreva as instruções para imprimir os campos que foram atribuídos na letra c.

44. Crie uma *struct* para descrever restaurantes. Os campos devem armazenar o nome do restaurante, o endereço, o tipo de comida (brasileira, chinesa, francesa, italiana, japonesa, etc) e uma nota para a cozinha (entre 0 e 5). Crie uma lista encadeada com esta *struct* e escreva um programa que: a) Insira um novo restaurante na lista; b) Leia uma lista de restaurantes a partir de um arquivo; c) Grave a lista de restaurantes para um arquivo; d) Liste todos os restaurantes na tela; e) Liste os restaurantes com cozinha com nota superior a um determinado valor, determinado pelo usuário; f) Liste todos os restaurantes com determinado tipo de comida, determinado pelo usuário.