Universidade Federal de Ouro Preto
Departamento de Computação e Sistemas – DECSI
Programação de Computadores I
Prof. Filipe Nunes Ribeiro

Lista de Exercícios 4 – Funções e procedimentos

Exercício 1 – Codifique uma função que receba por parâmetro a idade de uma pessoa, expressa em anos, meses e dias (todos inteiros), e retorne essa idade expressa em dias. Considere os meses como tendo 30 dias e desconsidere anos bissextos.

Exercício 2 – Analise o código abaixo e responde as seguintes questões.
a) Determine quais são as variáveis locais e globais deste programa, identificando a que função pertence cada variável local.
b) Mostre o que será impresso na tela do computador quando for executado este programa.

```c
#include <stdio.h>


int soma1(int q, int c);
int soma2(int ra);

int i = 10;
int j = 20;

int main()
{
    int i,k,ra,p;

    p = 10;
    ra = 5;

    for(i = 0; i< 3; i++)
    {
        k = soma1(ra, p);
        ra = soma2(k);
        printf("%d, %d\n",ra, k);
    }
    return 0;
}
int soma1(int q, int c)
{
    int soma = q+i+c;
    return soma;
}
int soma2(int ra)
{
    int k = j;
    ra = ra + k;
    return ra;
}
```

Exercício 3 – Crie um programa em C que peça ao usuário que digite um número inteiro e retorne a soma de todos os números de 1 até o número que o usuário introduziu ou seja: 1 + 2 + 3+ ... + n. Utilize uma função específica para calcular o somatório.

Exercício 4 – Codifique uma função que receba a média final de um aluno passado por parâmetro e retorne o seu conceito (através de uma variável char), conforme a tabela a abaixo. Crie um método main que solicite ao usuário que digite uma nova nota enquanto quiser continuar.

Nota	Conceito
De 0 a 49	D
De 50 a 69	C
De 70 a 89	B
De 90 a 100	A

Exercício 5 – Codifique um procedimento com a assinatura void estacao(int dia, int mes) que exiba no vídeo qual a estação do ano correspondente à data passada por parâmetro. Lembre-se que a primavera começa em 23 de setembro, o verão em 21 de dezembro, o outono em 21 de março e o inverno em 21 de junho. Crie um método main para testar a função estacao.

Ex:
estacao (2 5 , 1 0) ; /∗ Deve imprimir a mensagem : 25/10 e p r i m a v e r a . ∗/
estacao (2 9 , 1 2) ; /∗ Deve imprimir a mensagem : 29/12 e v e r a o . ∗/

Exercício 6 – Codifique uma função com a assinatura int contaimpar(int n1, int n2) que retorne o número de inteiros impares que existem entre n1 e n2 (inclusive ambos, se for o caso). Caso o valor de n2 seja menor que o de n1, a função deve tratar o intervalo como sendo de n2 até n1 sem que o invocador da função perceba.

Ex:

n = contaimpar (10 , 19) ; /∗ n recebe 5 (referente a : 1 1 , 1 3 , 1 5 , 1 7 , 1 9) ∗/
n = contaimpar (5 , 1) ; /∗ n recebe 3 (referente a : 1 , 3 , 5) ∗/

Exercício 7 – Codifique uma função com a assinatura int somaintervalo(int n1, int n2) que retorne a soma dos números inteiros que existem no intervalo fechado entre n1 e n2 (ou seja, incluindo n1 e n2). Caso o valor de n2 seja menor que o de n1, a função deve tratar o intervalo como sendo de n2 até n1 sem que o invocador da função perceba.
Ex:
n=somaintervalo (3 , 6) ; /∗ n recebe 18 (referente a : 3 + 4+ 5 + 6) ∗/
n=somaintervalo (5 , 5) ; /∗ n recebe 5 (referente a : 5) ∗/
n=somaintervalo (−2 ,3) ; /∗ n recebe 3 (referente a : −2 + −1 + 0 + 1 + 2 + 3) ∗/
n=somaintervalo (4 , 0) ; /∗ n recebe 10 (referente a : 4 + 3+ 2 + 1 + 0) ∗/

Exercício 8 – Crie uma função que exiba na tela os n (recebido por parâmetro) primeiros números da sequência de Fibonacci. Na matemática, a Sucessão de Fibonacci (também Sequência de Fibonacci), é uma sequência de números inteiros, começando normalmente por 0 e 1, na qual, cada termo subsequente corresponde à soma dos dois anteriores. 0,1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233...

Exercício 9 – Crie uma função em linguagem C chamado dado() que retorna, através de sorteio, um número inteiro de 1 até 6. Para gerar um número aleatório use a função rand como no utilizada no código abaixo (linha 8), que irá sortear exatamente um número entre 1 e 6. A função srand(time(NULL)) chamada na linha anterior é importante para definir uma semente que será a base para a geração dos números aleatórios. Neste caso usado no exemplo, a semente irá variar sempre, sendo a hora corrente da execução do programa definida em milissegundos, que é obtida por meio da função time(NULL). Se não utilizarmos a função srand o número gerado será sempre o mesmo.

Obs1: deve-se incluir a biblioteca stdlib.h para usar a função rand() e a biblioteca time.h para chamar a função time().

Obs2: a função srand precisa ser chamada apenas uma vez durante a execução, ou seja, a chamada a esta função não precisa estar dentro do método dado().

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include<time.h>
4
5   int main(){
6       int numero_sorteado;
7       srand(time(NULL));
8       numero_sorteado = (rand()%6)+1;
9       printf("Numero sorteado: %d\n", numero_sorteado);
10      return 0;
11  }
```

Exercício 10 – Use a função da questão anterior e lance o dado 1 milhão de vezes. Conte quantas vezes cada número saiu e exiba a porcentagem de cada um.