Tecnologia em Análise e Desenvolvimento de Sistemas - Estrutura de Dados
Lista de Exercícios para Estudos

PARTE I - funções e vetores

1) Simular o funcionamento (no papel) do seguinte programa. Quais valores são impressos na saída padrão?

```c
#include <stdio.h>
#include <stdlib.h>
void f1(int a){
   printf ("%d", a);
   a += a;
   printf ("%d", a);
}

int main() {
   int a = 2;
   int b = 3;
   f1 (a);
   f1 (b);
   printf ("%d %d", a, b);
   return 0;
}
```

2) Simular o funcionamento (no papel) do seguinte programa. Quais valores são impressos na saída padrão?

```c
#include <stdio.h>
#include <stdlib.h>
void f2(int a []){
   printf ("%d", a[1]);
   a[0]++;
   printf ("%d", a[0]);
}

int main() {
   int x [] = {3,2};
   f2(x);
   printf ("%d %d", x[0], x[1]);
   return 0;
}
```

3) Simular o funcionamento (no papel) do seguinte programa. Quais valores são impressos na saída padrão?

```c
#include <stdio.h>
#include <stdlib.h>
int f3(int a [], int b){
      int i;
      for (i = 0; i < b; i++){
            if (a[i] > 4){
                  a [i] += 2;
            }
            else{
                  a[i] -= 2;
            }
            a[i]++;
      }
      b++;
      return a[0] + a[1];
}
```

```
int main() {
	int x [] = {5, 6, 4, 1};
	int b = 4;
	int resultado = f3(x, b);
	printf ("%d %d %d %d %d %d %d", x[0], x[1], x[2], x[3], x[4], b, resultado);
	return 0;
}
```

4) Escrever uma função que recebe um valor real como parâmetro que representa uma temperatura em graus Fahrenheit e devolve este valor convertido para graus Celsius. Escrever outra função que recebe um valor real como parâmetro que representa uma temperatura em graus Celsius e devolve este valor convertido para graus Fahrenheit. Na função main, escrever um laço que permite ao usuário escolher que tipo de conversão deseja fazer. Este laço termina quando o usuário digitar a opção -1. Caso ele digite 1, realizar a conversão de Celsius para Fahrenheit usando a função escrita e mostrar o resultado. Caso o usuário digite 2, realizar a conversão de Fahrenheit para Celsius usando a função escrita e mostrar o resultado. **Somente a função main pode fazer chamadas às funções printf e scanf.**

As fórmulas para conversão são as seguintes:
°F = °C × 1.8 + 32
°C = (°F − 32) / 1.8

5)Escrever uma função com tipo de retorno float, chamada calculadora e com lista de parâmetros composta por um char e dois inteiros. Sua função deve utilizar uma estrutura switch/case para verificar qual o valor do char, que pode ser: +, -, * ou /. Como char é um tipo específico de inteiro, o switch/case pode ser usado naturalmente. Sua função deve fazer a operação matemática representada pelo símbolo contido no char, usando como operandos os dois inteiros recebidos como parâmetro e devolver o resultado. Cuidado com a divisão por zero. Escreva uma função main para testar a função calculadora com valores digitados pelo usuário.

6)Sabemos que no cabeçalho de função <stdlib.h> existe uma função chamada rand, que permite a geração de números pseudoaleatórios entre 0 e RAND_MAX, que é um número inteiro definido dentro do próprio cabeçalho, que para muitas implementações do compilador é 32768. Ou seja, a cada vez que seu programa chamar a função rand, que não recebe parâmetro algum, ela devolverá um número entre 0 e 32768. É natural a necessidade de números aleatórios para implementar um jogo, por exemplo. Mas muitas vezes é preciso limitar o intervalo dos números que são gerados. Por exemplo, se você quiser escrever um simulador de lançamento de dados, provavelmente vai querer utilizar números aleatórios entre 1 e 6, para representar cada face de um dado. Com algumas manipulações matemáticas, é possível escolher este intervalo. Considere a seguinte função, ainda sem implementação:

```
int gera_aleatorio (int n){

}
```

6.1 Implementar a função gera_aleatorio que devolve um número inteiro aleatório entre 0 e n-1. Dica: Considere o operador módulo.

6.2 Implementar a segunda versão de gera_aleatorio definida abaixo. Ela deve devolver um número inteiro entre primeiro e ultimo.

```
int gera_aleatorio2 (int primeiro, int ultimo){

}
```

7) Utilizando o conhecimento adquirido na questão 7, resolver os seguintes exercícios.

7.1) Implementar um simulador de lançamento de dados que executa o lançamento de um dados 6000 vezes. Imprimir uma tabela que mostra o número do lançamento e a face obtida. As faces são geradas randomicamente e variam entre 1 e 6. O simulador deve ser implementado por uma função chamada lança_dado.

7.2) Implementar um simulador que executa o lançamento de um dado e de uma moeda 10000 vezes. O simulador deve ser implementado por uma função chamada lança_dado_e_moeda. As faces do dado podem variar de 1 a 6 e os números representando cara ou coroa são 0 ou 1 respectivamente. Imprimir uma tabela que mostra o número do lançamento seguido da face do dado obtida seguido de cara ou coroa.

8) Um jogo de dados chamado craps funciona da seguinte forma:

Um jogador joga dois dados. Cada dado tem seis faces. As faces contém os números 1, 2, 3, 4, 5 e 6. Depois de feito o lançamento simultâneo dos dois dados, a soma das duas faces que ficaram para cima é calculada. Se a soma é igual a 7 ou 11 no primeiro lançamento, o jogador vence. Se a soma é 2, 3 ou 12 no primeiro lançamento (fenômeno chamado de craps), o jogador perde. Se a soma é 4, 5, 6, 8, 9 ou 10 no primeiro lançamento, esta soma se torna a pontuação do jogador. Para vencer, é preciso continuar lançando os dados até que se repita essa pontuação. Por exemplo, se a pontuação do usuário for 10, ele continua lançando os dados até obter um novo 10, momento em que vence o jogo. Mas ele perde caso obtenha um 7.

Utilizar o paradigma da divisão e conquista para implementar um simulador para este jogo, utilizando funções. Ou seja, identificar pequenas partes que compõem este problema e escrever uma função que resolve cada pequena parte. Esta é a parte da divisão. Na função main, chamar as funções adequadamente, unindo-as de forma a fazer o simulador funcionar adequadamente. Esta é a parte da conquista.

9)Considere o seguinte programa:

```
void f1 (){
      f2();
}

void f2(){
      f1();
}

int main() {
      f1();
}
```

Fazer um estudo sobre o programa e, sem executá-lo no computador, explicar o problema existente.

PARTE II - structs

10) Considere a estrutura definida a seguir, que representa um ponto no plano cartesiano:

```
typedef struct ponto {
   float x;
   float y;
} t_ponto;
```

a partir dessa estrutura é possível definir um círculo com um ponto central e um raio.

10.1) definir a estrutura círculo como descrito acima
10.2) criar uma função que recebe 2 pontos e devolve a distância entre eles $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
10.3) criar uma função que recebe um círculo e um ponto e devolve se o ponto pertence ou não ao círculo

11) Agora, considere a estrutura definida a seguir, que representa um número racional ou fração:

```
typedef struct fracao {
   int num;
   int den;
} t_fracao;
```

11.1) criar uma função que recebe 2 frações e devolve qual a maior (a primeira ou a segunda)
11.2) criar uma função que recebe 2 frações e devolve as duas com o mesmo denominador (MMC)

PARTE III - pilhas

12) Suponha uma pilha P para a qual tenham sido executadas 25 inserções, 12 exibições de topo, 10 remoções, das quais 3 geraram resultado vazia.
12.1) Qual a posição do topo ao final das operações?
12.2) Escrever uma função para exibir a posição e o elemento do topo de uma pilha

13) Considere uma pilha vazia P, qual sua configuração após as operações: empilha o 5, empilha o 3, desempilha, empilha o 2, empilha o 8, desempilha, desempilha, empilha o 9, empilha o 1, desempilha, empilha o 7, empilha o 6, desempilha, empilha o 10, desempilha, empilha o 4, desempilha, desempilha.

14) Construir uma função para testar se uma pilha P1 tem mais elementos que uma pilha P2. Considere que P1 e P2 já existem.

15) Construir uma função para inverter a posição dos elementos de uma pilha P. Você pode criar pilhas auxiliares, se necessário. Mas o resultado precisa ser dado na pilha P.

16) Construir uma função que verifica se duas pilhas P1 e P2 são iguais. Duas pilhas são iguais se possuem os mesmos elementos, na mesma ordem. Você pode utilizar pilhas auxiliares também, se necessário.

17) Construir uma função que retorna o valor do elemento do topo de uma pilha, sem desempilhar.

PARTE IV - filas

18) Considere uma fila F vazia para a qual foram executadas 32 inserções, 15 remoções, 5 das quais geraram fila vazia. Qual o tamanho atual de F?

19) Se a fila da questão anterior foi implementada com um vetor de tamanho 30, sem nunca ter dado fila cheia, quais os possíveis valores de prim e ult?

20) Escrever uma função para exibir o primeiro da fila: posição e elemento.

21) Escrever uma função para exibir o último de uma fila: posição e elemento.

22) Escrever uma função que devolve o comprimento (ou seja, o número de elementos) de uma fila dada.