Métodos e Técnicas de Programação
2ª Prova (2017/2)

- Turma V

1. João Víctor Oliveira Mendes 11711EBI005
2. Mariana Rigo Estevão 11711EBI008
3. Felipe Lobato Ferreira 11711EBI026

QUESTÃO 01

MAT0: 11711EBI005	MAT1: 11711EBI008	MAT1: 11711EBI008
KANO0 : 3	KCUR0 : 2	KNUM0 : 6
KANO1 : 3	KCUR1 : 2	KNUM1 : 9
KANO2 : 3	KCUR2 : 2	KNUM2 : 9

QUESTÃO 02

A) A força gravitacional entre as esferas azul e vermelha, distantes 15,704299 metros uma da outra, é de 7,302179.

B) Inicialmente, é definido uma struct strEsfera, composta por 4 variáveis (as coordenadas x, y e z e a massa) chamada Esfera.

Na main, são definidas duas Esferas, chamadas azul e vermelha. De acordo com os resultados obtidos no exercício anterior, são definidas as coordenadas (x,y,z) e as massas de ambas as esferas.

São definidas, além da main, duas funções.

A primeira, chamada “distancia”, recebe como parâmetros as Esferas azul e vermelha e, a partir de suas coordenadas, faz o cálculo da distância das mesmas. Primeiramente, são realizadas as operações de diferença entre cada coordenada das esferas e o resultado e elevado ao quadrado. Os três valores obtidos (quadrado da diferença das coordenadas x, das coordenadas y e das coordenadas z) são somados e a raiz dessa soma é o resultado final da função (representa a distância entre as esferas).

A segunda função, forcaGravitacional, recebe os mesmos parâmetros (as duas esferas -massa e coordenadas) e calcula a força gravitacional. Para a realização deste cálculo, a função anterior (“distancia”) é chamada e seu resultado é utilizado na conta da força, que dá-se pela divisão do produto entre as duas massas e e a constante de gravitação universal pelo quadrado da distância (proveniente da primeira função). A constante, aqui, foi definida como variável global de valor 6.67e-11.

Ambas funções são chamadas dentro do comando printf presente na main, que mostra ao usuário tanto a distância como a força gravitacional encontrados.

QUESTÃO 03

A) Instancia: zvrnjfb^ 1,700000
Instancia: yuqmiea] 1,700000
Novainsta: yuqmiea] -8,300000

B) 18 bytes (10 bytes provenientes da string codigo e 8 bytes da variável preço).

C) É definido, inicialmente, uma struct stArmazem composta pela string código, de tamanho KNUM1 (que vale 9, de acordo com o encontrado em exercicios anteriores) e o preco. É definido, então, Armazem.

Na main, são definidas as variáveis “instancia” e “novainsta” do tipo Armazem.

Posteriormente, é feito um for que “varre” a string “codigo” de “instancia” definindo cada caracter da mesma, que é calculado pela seguinte operação: ‘z’ (122 na tabela ASCII) menos a multiplicação de i(que varia de 1 até 9(KNUM1) e representa a posição atual que está sendo varrida da string) por ((KANO1 - KCUR1 + KNUM2 + KANO0), que, no nosso caso, vale 4).
Depois de definir os elementos de cada posição da string, a última posição recebe ‘\0’ (para indicar onde está seu fim).
o preco da instancia é definido pela divisão de 17 por 10 (sendo 17 o valor da soma de KANO2, KNUM2, KCUR2 e KANO0).
E feita, então, a impressão dos resultados obtidos pelos dois cálculos acima.
Em seguida, define-se que a novainsta recebe o resultado da função “criaInstancia”.
Essa função, por sua vez, recebe como parâmetros o codigo e o preco da “instancia”, sendo que o codigo recebido é utilizado pela nova função como um ponteiro. Dentro de “criaInstancia” é definida a variável temp, também do tipo Armazem, e i, do tipo inteiro. Utilizando o comando for e a variável i que varia de 1 até 9, é feita a varredura da string codigo e, para cada posição, é realizado o seguinte comando:
Se a posição não for igual a 0, seu valor seu valor é decrementado
Caso contrário, ela permanece com o mesmo valor.
Nesse caso, tem-se que essas mudanças estão ocorrendo na string codigo da variavel “instancia”, passado como ponteiro. Posteriormente, a string codigo da variável temp recebe esses mesmos valores, recém definidos.
Assim, com o laço finalizado, faz-se o cálculo do preco, que e decrementado em 10 unidades e passa a representar, também, o preco da variável temp. A função retonar como resultado temp.

A main recebe, então, temp.codigo e temp.preco, que serão o novo codigo e preco da variavel novainsta.
Os codigos e os preços, tanto de instancia como de novainsta, são mostrados ao usuário através de dois comandos de printf.
D) A função criaInstancia modifica o código da variável instancia e define o código da variável novainsta como sendo igual ao anterior. Portanto, ao final temos que o preço inicial de instancia permanece o mesmo (diferente do calculado para novainsta). Já o código de instancia inicial foi modificado, e é igual código de novainsta.

QUESTÃO 04

A)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#define N 3
typedef
struct stTexto
{
int id;
char mat[12];
}
Texto;

void imprime(Texto * dado, int qtde)
{
if(qtde > 0)
{
imprime(dado+1,qtde-1);
printf(": (%d) %s ", dado->id, dado->mat);
}
else printf(":\n");
}

int main()
{
Texto grupo[N];
int i;
for(i = 0; i < N; i++)
        grupo[i].id = i;
strcpy(grupo[0].mat,"11711EBI005");
```

```
strcpy(grupo[1].mat,"11711EBI008");
strcpy(grupo[2].mat,"11711EBI026");
imprime(grupo, N);
getche();
return 0;
}
```

RESULTADO:

```
(2) 11711EBI026
(1) 11711EBI008
(0) 11611EBI005
```

B)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#define N 3
typedef
struct stTexto
{
int id;
char mat[12];
}
Texto;

void imprime(Texto * dado, int qtde)
{
if(qtde > 0)
{
printf(": (%d) %s ", dado->id, dado->mat);
imprime(dado+1,qtde-1);
}
else printf(":\n");
}
int main()
{
Texto grupo[N];
int i;
for(i = 0; i < N; i++)
        grupo[i].id = i;
strcpy(grupo[0].mat,"11611EBI005");
strcpy(grupo[1].mat,"11711EBI008");
strcpy(grupo[2].mat,"11711EBI026");
imprime(grupo, N);
```

```
getche();
return 0;
}
```

RESULTADOS:

(0) 11611EBI005
(1) 11711EBI008
(2) 11711EBI026

C) Para que o resultado mostrado ao usuário seja impresso em ordem crescente, é preciso garantir que a impressão seja feita antes da recursão presente na função imprime. Isso porque essa função recebe as matrículas e N (quantidade de matrículas).

Como o grupo[0] é o primeiro a entrar na função, devia à presença da recursão antes da impressão do mesmo, ocorre primeiro a realização da função para grupo[1]. Por sua vez, antes de realizar a impressão de grupo[1], devido à recursão antes do printf, é feita a função imprime para o grupo[2].

Ou seja, para colocarmos em ordem crescente, é necessário que o printf esteja antes da recursão, pois, desse modo, a função imprime primeiramente o valor da matrícula no grupo[0], para depois, com a recursão, passar aos grupos posteriores, até o 2.

QUESTÃO 05

A) Fibonacci (5) = 5
Fibonacci(8) = 21
Fibonacci(8) = 21
Numero secreto = 133

B)

C)

id escopo	Escopo origem	Retorno	Índice	Condição	Resposta
0	main		6	falsa	fib(5)+fib(4)
1	0		5	falsa	fib(4)+fib(3)
2	1		4	falsa	fib(3)+fib(2)

3	2		3	falsa	fib(2)+fib(1)
4	3		2	falsa	fib(1)+fib(0)
5	2		1	verdadeira	1
6	2		0	verdadeira	0
4	3	5 e 6	2	falsa	0 + 1
3	2	4	3	falsa	1 + fib(1)
2	1	3	4	falsa	2 + fib(2)
1	0	2	5	falsa	3 + fib(3)
0		1	6	falsa	5 + fib(4)
		0			8

QUESTÃO 06

```
int trib (int * N)
{
    int num;
    num = (N>2)? trib(N-1)+2*trib(N-2)+3*trib(N-3) : N;
    return num;

}
```