DANILO DE NADAI SICARI – LISTA 1 – ESTRUTURA DE DADOS

PONTEIROS

1.Quais serão os valores de x, y e p ao final do trecho de código abaixo?

```
int x, y, *p;
y = 0;
p = &y;
x = *p; x = 0
x = 4; x = 4
(*p)++; *p = 1
--x; x = 3
(*p) += x; *p = 4
```

X = 3

Y = 4

*p = 4

2.Os programas (trechos de código) abaixo possuem erros. Qual(is)? Como deveriam ser?

a)

```
void main()
 {
int x, *p;
x = 100;
p = &x;
printf("Valor de p: %d.\n", *p);
}
```

Podemos ou não chamar de erro este caso. Como p foi declarado como ponteiro, temos duas situações:

1 – p = x, como seria o código original e então p apontaria para o endereço 100

2 – p = &x, como sugerido acima, onde p apontaria para o endereço da variável x

b)

```
void troca (int *i, int *j)
 {
int *temp;
*temp = *i;
*i = *j;
*j = *temp;
```

}

Poderiamos destacar que esta função, a qual recebe 2 ponteiros como referencia, declara um terceiro ponteiro *temp, porem não o assume a qualquer endereço de memoria. Como tem o objetivo de trocar dois valores, poderíamos apenas declarar temp como uma variavel inteira para auxiliar a execução da função.

```
void troca (int *i, int *j)
 {
int temp;
temp = *i;
*i = *j;
*j = temp;
}
```

c)

```
char *a, *b;
a = "abacate";
b = "uva”;
if (a < b)
printf ("%s vem antes de %s no dicionário", a, b);
else
printf ("%s vem depois de %s no dicionário", a, b);
```

Estao sendo declarados 2 ponteiros, o quais não assumem nenhum endereço. Poderia neste caso não se trabalhar com ponteiros e apenas com dois arrays do tipo char, declarando-os de forma apropriada e comparando seu primeiro caracter para saber a ordem das palavras. Seria mais apropriado comparar todos os caracteres da palavra para não correr o risco de ordenar palavras iguais e ou semelhantes.

```
int main(void)
{
        char a[20], b[20];
        //a = "abacate";
        memcpy(a, "abacate", sizeof("acabate"));
        //b = "uva";
        memcpy(b, "uva", sizeof("uva"));
        if (a[0] < b[0])
```

```
                printf ("%s vem antes de %s no dicionario", a, b);
        else
                printf ("%s vem depois de %s no dicionario", a, b);
        return 0;
}
```

Para representar o mesmo código, utilizando ponteiros em C:

```
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
        //Cria apontador para 20 espacos de memoria de 1 byte
        char *a = malloc(sizeof(char)*20);
        char *b = malloc(sizeof(char)*20);

        if (a == NULL || b == NULL)
        {
                printf("Memoria insuficiente");
                exit(1);
        }
        else
        {
                //*a = "abacate";
                memcpy(a, "abacate", sizeof("abacate"));
                //*b = "uva";
                memcpy(b, "uva", sizeof("uva"));
                if (a[0] < b[0])
                        printf ("%s vem antes de %s no dicionario", a, b);
                else
                        printf ("%s vem depois de %s no dicionario", a, b);
```

```
        }

        return 0;
}
```

3)Suponha que os elementos do vetor v são do tipo int e cada int ocupa 8 bytes no seu computador. Se o endereço de v[0] é 55000, qual o valor da expressão v + 3?

Seria: 55000 + 3*8 = 55024

4)Escreva uma função mm que receba um vetor inteiro v[0..n-1] e os endereços de duas variáveis inteiras, digamos min e max, e deposite nessas variáveis o valor de um elemento mínimo e o valor de um elemento máximo do vetor. Escreva também uma função main que use a função mm.

5)Suponha que v é um vetor. Descreva a diferença conceitual entre as expressões v[3] e v + 3.

v[3] aponta para o conteúdo do vetor v, 4a posição.

v + 3 ira retornar o endereço de v + 3 posicoes, no caso a quarta posição.

6)(sem usar o computador) Qual o conteúdo do vetor a depois dos seguintes comandos.

```
int a[99];
for (i = 0; i < 99; ++i)
        a[i] = 98 - i;
// a[0] = 98 – 0 = 98
// a[1] = 98 – 1 = 97
// a[2] = 98 – 2 = 96
// ...
// a[98] = 98 – 98 = 0
for (i = 0; i < 99; ++i)
        a[i] = a[a[i]];
// a[1] = 97, logo: a[1] = a[a[1]] >> a[1] = a[97]  >> a[1] = 1
// a[2] = 96, logo: a[2] = a[a[2]] >> a[2] = a[96]  >> a[2] = 2
```

// ...

// a[98] = 0, logo: a[98] = a[a[98]] >> a[98] = a[0] >> a[98] = 98

O programa insere valores de 98 a 0 no vetor, pelo primeiro loop. Já no segundo loop o vetor e invertido, tendo valores de 0 a 98.

7)Escreva uma função chamada troca que troca os valores dos parâmetros recebidos. Sua assinatura deve ser:

void troca(float *a, float *b);

8)Crie uma função que receba uma string como parâmetro (de tamanho desconhecido) e retorne uma cópia da mesma. A assinatura da função deve ser: char *strcopy(char *str);

//Programa brica com espaços...

9)Escreva uma função que recebe como parâmetros um vetor de inteiros v, o número de elementos dele N e ponteiros para variáveis nas quais devem ser armazenados os valores maximo e minimo do vetor. Sua assinatura deve ser: void maximoMinimo(int *v, int N, int *maximo, int *minimo);

10) Qual o resultado do código abaixo? Explique cada linha.

int x = 100, *p, **pp; //x recebe valor 100, *p é um apontador para inteiro e **p é um apontador para apontador para inteiro

p = &x; //apontador p aponta para endereço de x, *p = 100

pp = &p; //apontador pp aponta para apontador p o qual aponta para endereço de x, *pp = 100

printf(“Valor de pp: %d\n”, **pp); //imprime o valor de pp, cujo é 100

STRUCTS

1. Escrever um programa que cadastre o nome, a matrícula e duas notas de vários alunos. Em seguida imprima a matrícula, o nome e a média de cada um deles.

2.Escrever um programa que cadastre o nome, a altura, o peso, o cpf e sexo de algumas pessoas. Com os dados cadastrados, em seguida localizar uma pessoas através do seu CPF e imprimir o seu IMC.

3. Escrever um programa que cadastre vários produtos. Em seguida, imprima uma lista com o código e nome da cada produto. Por último, consulte o preço de um produto através de seu código.

4. Escreva um programa que simule contas bancárias, com as seguintes especificações:

- Ao iniciar o programa vamos criar contas bancárias para três clientes.
    - Cada conta terá o nome e o CPF do cliente associado a ela.
    - No ato da criação da conta o cliente precisará fazer um depósito inicial.
- Após as contas serem criadas, o sistema deverá possibilitar realizações de saques ou depósitos nas contas.
    - Sempre que uma operação de saque ou depósito seja realizada, o sistema deverá imprimir o nome do titular e o saldo final da conta.