Algoritmos e Fundamentos da Programação II

IFTM - *Campus* Ituiutaba

Ciência da Computação – 2º Sem. 2024

Prof. Alencar Melo Jr., Dr. Eng.

Lista de exercícios 7
Assunto: Ponteiros

1. Para fazer a leitura de uma variável num, a função scanf se refere a mesma da seguinte forma: &num. Explique.

2. Escreva uma função do tipo void que recebe como parâmetro um inteiro e o eleva ao quadrado. Mostre como esta função poderá ser chamada na função main.

3. Suponha que um vetor de inteiros como parâmetro de uma função. Mostre como a passagem deste parâmetro pode ser feita por referência e por valor.

4. Quais das seguintes instruções é correta para declarar um ponteiro?

(a) int _ptr x;
(b) int *ptr;
(c) *int ptr;
(d) *x;

5. Qual é a maneira correta de referenciar o conteúdo de ch, assumindo que o endereço de ch foi atribuído ao ponteiro indica?

(a) *indica;
(b) int *indica
(c) ch;
(d)*ch;

6. Na expressão float *fptr, o que é do tipo float?

(a) A variável fptr
(b) O endereço de fptr
(c) A variável apontada por ftpr
(d) Nenhuma das anteriores

7. Assumindo que o endereço do variável var foi atribuído a um ponteiro pointvar, escreva uma expressão que não usa var (acesso indireto) e divida var por 10.

8. Assumindo que o endereço de uma variável inteira vox foi atribuído a uma variável ponteiro invox, quais das seguintes expressões são verdadeiras?

(a) vox ==&invox
(b) vox==*invox
(c) invox==*vox
(d) invox==&vox

9. Qual é a instrução que deve ser adicionada ao programa seguinte para que ele trabalhe corretamente?

```
main( )
{
    int j, *ptrj;
    *ptrj = 3;
    ...
}
```

10. Assumindo que queremos ler o valor de x e o endereço de x foi atribuído a ptrx, a instrução seguinte é correta? Justique. Instrução: scanf("%d", *ptrx);

11. Seja o seguinte trecho de programa:

```
int i=3, j=5;
int *p, *q;
p= &i;    q= &j;
```

Qual é o valor das seguintes expressões?

(a) p==&i
(b) *p-*q
(c) **&p
(d) 3*-*p/(*q)+7

12. Qual será a saída deste programa supondo que i ocupa o endereço 4094 na memória?

```
main( )
{
    int i=5, *p;
    p= &i;
    printf(" %u %d %d %d \n", p, *p+2, **&p, 3**p, **&p+4);
}
```

13. Se i e j são variáveis inteiras e p e q ponteiros para int, quais das seguintes expressões de atribuições são ilegais?
    (a) p= &i;
    (b)*q= &j;
    (c)p= &*&i;
    (d) i= (*&)j;
    (e) i = *&j
    (f) i= *&*&j;
    (g) q= &p;
    (h) i= (*p)++ + *q;

14. O seguinte programa tem um erro de conceito. Qual é?

```
#define NUMERO 987
main( )
{
        int *p= &NUMERO;
        printf("Numero = %d \n", *p);
}
```

15. Assumindo que pulo[ ] é uma matriz de uma dimensão (vetor) do tipo int, quais das seguintes expressões referenciam o valor do terceiro elemento da matriz ?

(a) *(pulo +2)
(b) *(pulo + 4)
(c) pulo + 4
(d) pulo + 2

16. Supor a declaração:

```
int mat[4], *p, x;
```

Quais expressões são válidas? Justifique.
(a) p = mat + 1;
(b) p = mat++;
(c) p = ++mat;
(d) x = (*mat)++;

17. O que fazem os seguintes programas quando executados?

(a)

```
main( )
{ int mat[ ] = {4, 9, 13};
int j;
for (j= 0; j < 3; j++)
   printf("%d ", *(mat + j));
}
```

(b)

```
main( )
{ int mat[ ] = {4, 9, 13};
 int j;
 for (j= 0; j < 3; j++)
     printf("%d ", mat + j);
 }
```

(c)

```
main( )
{ int mat[ ] = {4, 9, 13};
int j;
for (j= 0; j < 3 ; j++)
     printf("%d ", *mat + j;
 }
```

18. O que faz o programa seguinte quando executado?

```
main( )
{
```

```
        int mat[ ] = {4, 9, 12};
        int j, *ptr;
        ptr= mat;
        for (j= 0; j < 3; j++)
          printf("%d", *ptr++);
}
```

O último comando poderia ser substituído por printf("%d", *mat++); ?

19. O que faz o programa seguinte quando executado?

```
main( )
{
        int mat[ ] = {4, 9, 12};
        int j, *ptr;
        ptr= mat;
        for (j= 0; j < 3; j++)
                printf("%d", (*ptr)++);
}
```

20. Seja vet um vetor de 4 elementos: TIPO vet[4]. Supor que depois da declaração, vet armazena o endereço de memória 4092 (ou seja, o endereço de vet[0], supondo que a memória endereça bytes). Supor também que na máquina usada uma váriável do tipo char ocupa 1 byte, do tipo int ocupa 2 bytes, do tipo float ocupa 4 bytes e do tipo double ocupa 8 bytes.
    Qual o valor de vet +1, vet +2 e vet +3 se:
    (a)vet for declarado como *char*?
    (b)vet for declarado como *int*?
    (c)vet for declarado como *float*?
    (d)vet for declarado como *double*?

Bom Trabalho!