Unidade 00b- exercícios

1 e 4- resposta em arquivo .c

5- O codigo verifica se o numero digitado é 65, 69, 73, 79, 85, 97, 101, 105, 111, 117 e, se verdadeiro, retorna true.

6- Faz o mesmo que no anterior, e logo verifica se o char inserido é consoante e se el é vogal.

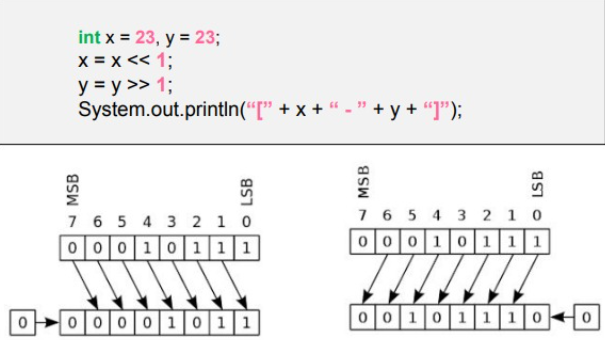
7- duvidas

8- O primeiro é mais simples de entender

9- O primeiro método realiza a subtração após seu comando, enquanto no segundo método, o i já é subtraído no momento da leitura dessa linha de código.

10- O programa mostra na tela os valores de cada variável aumentados a um enquanto a condição do while for verdadeira (1111, 2222, 3333, etc)

11- O código imprime 46-11 pois x=x<<z; significa que o valor de x foi movimentado para a esquerda em 1 bit (e os novos são preenchidos por 0), o que multiplica o número por 2 elevado a z (nesse caso, x=x<<1, então 23x2 = 46). Enquanto isso, y=y>>1; é o deslocamento do valor de y para a direita em 1 posição (1 bit), o que é o mesmo que dividi-lo por 2 levado a 1 (divisão inteira de 23/2 = 11).



00h Recursividade

1-A condição de parada no primeiro código é quando a fatoração chega no momento em que n=1 e sua chamada recursiva é na multiplicação do n com a mesma função (com um novo valor de n-1). No segundo código, a condição de parada é quando n chegar ao valor 0 ou 1, e sua chamada recursiva é ao somar a função com o valor n-1 e com o valor n-2.

2-O código printa 2 por ter o primeiro print do valor referenciado sem alterações, depois 1 pois a condição do if é aceita, printa 0 por ser aceita e é printado 0 novamente por não ser aceita a condição. Assim, volta-se a aceitação da primeira condição onde i=1 e onde i=2, printando 1 e 2 em seguida.

3- O programa printa:

main – inicio

1o – inicio

2o - inicio e fim

1º - fim

main – fim

00i Ponteiros

1- ~~1º significa a criação de um ponteiro para um array de variáveis inteiras. 2ºsignifica a criação de um objeto a ser alocado.~~ Corrigindo:

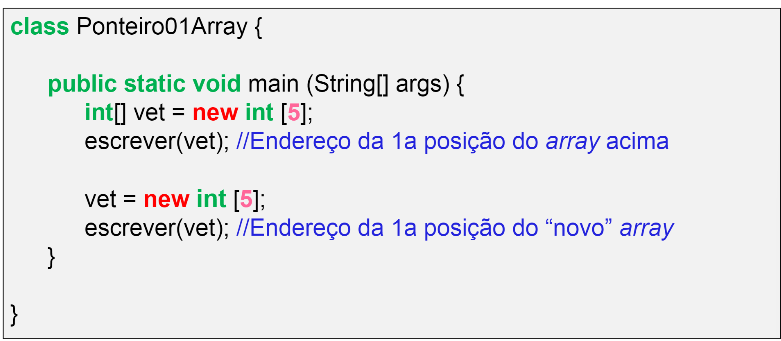
int [ ] vet 🡪 Cria-se um ponteiro com o nome vet

= new int [5] 🡪 Aloca-se 5 espaços de memória para armazenar números inteiros

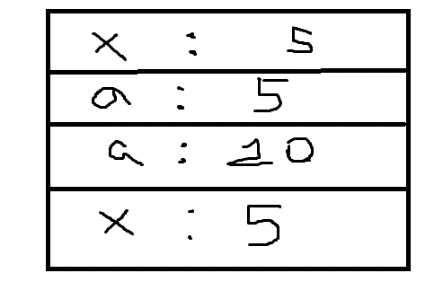
int [ ] vet = new int [5] 🡪 Aponta-se o ponteiro criado para a área de memória alocada.

Assim, o vet armazena o endereço de memória da primeira posição do array, não o conteúdo das posições 0 a 4 em si.

2- O programa escreve o endereço de memória da primeira posição alocada para o array criado e depois imprime o mesmo para o novo array.



3-



|  |
| --- |
| y[0]: 0 |
| y[1]: 1 |
| y[2]: 2 |
| y[3]: 3 |
| y[4]: 4 |
| b[0]: 0 |
| b[1]: 5 |
| b[2]: 10 |
| b[3]: 15 |
| b[4]: 20 |
| b[0]: 0 |
| b[1]: 1 |
| b[2]: 2 |
| b[3]: 3 |
| b[4]: 4 |
| (Sobrescreveu o y[] como b[]) |
| y[0]: 0 |
| y[1]: 5 |
| y[2]: 10 |
| y[3]: 15 |
| y[4]: 20 |

4-

5- Cliente c 🡪 ponteiro criado chamado c.

= new Cliente(); 🡪 objeto criado com o tipo da classe Cliente.

Cliente c = new Cliente(); 🡪 aponta-se o ponteiro para o objeto (o c não é um objeto, mas sim um ponteiro que armazena o endereço de memoria do objeto!)

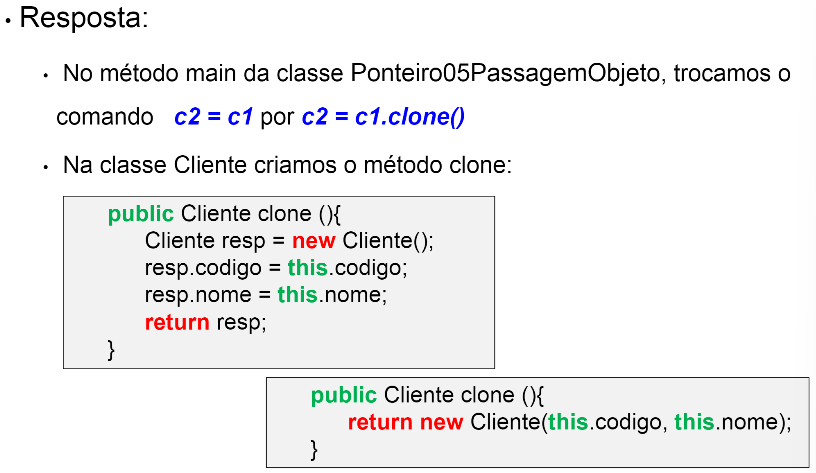
6- Tela

|  |
| --- |
| ADDRs:\nc1(null)\nc2(null)\nc3(null) |
| ADDRs:\nc1(7Ah)\nc2(7Ah)\nc3(A5h) |
| ATRIBUTOS |
| c1(3/aa) |
| c2(3/aa) |
| c3(2/bb) |

7- Memoria

|  |
| --- |
| 1/aa – endereco c1 |
| endereco c1 = c2 |
| 2/bb – endereco c3 |
| 4/dd – endereco c1 (mudou) |
| ? |

8- Para fazer com que os atributos de um objeto sejam iguais aos de um outro objeto apontado por outro ponteiro(mas em objetos diferentes, sem ser c1=c1), adiciona-se o .clone()



9- Tela

|  |
| --- |
| ADDRs:\nc1(null)\nc2(null)\nc3(null) |
| ADDRs:\nc1(7Ah)\nc2(8Ah)\nc3(A5h) |
| ATRIBUTOs: |
| c1(1/aa) |
| c2(3/aa) |
| c3(2/bb) |

10- Tela

|  |
| --- |
| endereço/1/aa |
| endereço/null/null |
| endereço/null/null |
| endereço/null/null |
| endereço/null/null |
| endereço/null/null |

11- ? porque diferentes?

12- verdadeiras: I, II 🡪 Letra a)

OO – Orintação a Objetos

1-

|  |
| --- |
| endereço |
| Outro endereço |
|  |
|  |