**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**

Instituto de Ciências Exatas e de Informática

Algoritmos e Estruturas de Dados 1 (AEDS 1)

Profa.: Rosilane Mota

**Lista de Exercícios – Ponteiros**

1. Explicar cada um e dar a diferença entre: p++; (\*p)++; \*(p++);

R: O p++ é o mesmo que dizer que p recebe p mais um, ou seja p = p + 1;. É usado para acrescentar uma unidade no valor total de uma variável qualquer, é bastante usado para laços de repetição que variam de 1 em 1. Já a expressão com (\*p)++ é o mesmo que dizer que o valor que está no endereço de p recebe endereço de p mais um. É bem parecido com a primeira opção, visto que altera o valor da variável em uma unidade, porém nesse caso estamos alterando esse valor por meio de um ponteiro que aponta o endereço de p no qual o valor deve ser incrementado em uma unidade. Por fim no caso de \*(p++); a gente incrementa um no endereço para qual o ponteiro p aponta, ou seja, se antes o endereço era 1 após realizar esse comando o endereço passa a ser 2. Nesse caso o valor que estava no endereço 1 não muda, apenas o valor do endereço que o ponteiro armazenava vai se alterar e partir de agora esse ponteiro apontar para outro local.

No caso de \*(p+10); o endereço do ponteiro p não vai se alterar, porém, o valor do endereço vai receber o que estiver no endereço p+10, ou seja, o valor que o ponteiro p aponta vai se alterar para um valor de um endereço maior por dez unidades. No caso, se o endereço do ponteiro p for A e seu valor 1, após realizar isso o endereço continuaria sendo A, mas com um valor diferente do 1 que ali estava.

1. Se o endereço de uma variável valor foi atribuído a um ponteiro valorPtr, quais alternativas são verdadeiras? Justifique sua resposta.

R: Apenas a opção C e D. Quando especificamos o sinal de ‘&’ quer dizer que estamos nos referindo ao endereço daquela variável, e quando apenas colocamos o nome do ponteiro sem o ‘\*’ estamos nos referindo ao endereço que aquele ponteiro armazena e não ao seu valor, logo, valorPtr é igual a &valor (C). Ao apenas usarmos o nome da variável sem o acréscimo de nada em sua volta estamos acessando o seu valor interno e ao usarmos o ‘\*’ antes de um ponteiro estamos dizendo que queremos acessar o valor mantido no endereço que aquele ponteiro aponta. Logo, valor é igual a \*valorPtr (B).

1. Observe os dois programas a seguir, Código 1 e Código 2. Qual. É a diferença entre eles? Qual é o valor impresso para ptr em cada um dos códigos? Porque?

R: Ambos os códigos apresentam o ponteiro ptr e um laço de repetição para alterar seu valor dinamicamente. O código 1 apresenta internamente dentro do for uma estrutura que a cada repetição do laço incrementa um no valor que o ponteiro aponta, ou seja, o valor se incrementa a cada laço de repetição, por isso no final o resultado é 15, pois no início o valor do endereço que ptr apontava era 10 e após o laço de repetição de cinco vezes o valor final é 15. O código 2 apresenta internamente a estrutura faltando o ‘\*’ dentro do laço de repetição, logo, com cada nova repetição do for o que vai ser alterado é o endereço para qual o ponteiro aponta. Logo, se no início o ponteiro apontava para o endereço 0xc9e264 ao fim de cinco laços de repetição vai apontar para o endereço 0xc9e274.

10.

|  |  |  |  |
| --- | --- | --- | --- |
| x | y | px | py |
| 5 | 6 | 0x7ffd720f1b38 | 0x7ffd720f1b34 |
| 5 | 6 | 0x7ffd720f1b34 | 0x7ffd720f1b34 |
| 5 | 36 | 0x7ffd720f1b34 | 0x7ffd720f1b34 |
| 5 | 38 | 0x7ffd720f1b34 | 0x7ffd720f1b34 |