ORIENTAÇÃO A OBJETOS

SISTEMAS DE INFORMAÇÃO

DR. EDNALDO B. PIZZOLATO

GERENCIAMENTO DINÂMICO DE MEMÓRIA (C++)

Tanto em Java como em C++ é possível dimensionar o espaço utilizado em memória através da alocação (em tempo de execução) do recurso necessário.

Em C++ o programador é responsável por devolver o recurso solicitado.

Em Java, isso fica transparente para o programador.

Em C++ você deverá fazer uso dos operadores new e delete.

Com Java, você poderá utilizar new.

o operador **new** (C++)

- Exemplo:

```
Hora *HoraPtr;
HoraPtr = new Hora;
```

- **new**
  - Cria objeto do tamanho apropriado
    - Erro se não houver espaço em memória
  - Chama construtor padrão (default)
  - Retorna ponteiro do tipo especificado

o operador **new** (C++)

- Inicialização

```
double *ptr = new double( 3.14159 );
Hora *HoraPtr = new Hora( 12, 0, 0 );
```

- Alocando arrays

```
int *meuArray = new int[ 10 ];
```

o operador `delete` (C++)

- Destrói dinamicamente objetos alocados (libera espaço)
- Exemplo:

```
delete HoraPtr;
```

- Operador **`delete`**
  - Chama destrutor para o objeto
  - Libera memória associada ao objeto
    - Pode ser reutilizado
- Delete para arrays

```
delete [] meuArray;
```

ESTUDO DE CASO
CLASSE VETOR

```
class VETOR {
public:
      VETOR( int = 10 );              // constructor padrão
      ~VETOR();                       // destrutor
      int getTam() const;             // tamanho
private:
      int tam; // tamanho
      int *ptr; // ponteiro para o primeiro elemento do
   array
};
```

```
VETOR::VETOR( int T )
{
      // valida o VETOR
      tam = ( T > 0 ? T : 10 );
      ptr = new int[ tam ]; // aloca o vetor
      for ( int i = 0; i < tam; i++ )
         ptr[ i ] = 0;   // inicializa o vetor
}
```

Construtor padrão

```
VETOR::~VETOR( )
{
      delete [] ptr;
}
```

Destrutor devolve a memória

Conclusões:

Usar new e delete é semelhante a utilizar malloc, calloc e free, cfree.

O mais importante é saber como o processo de alocação acontece e nunca esquecer de devolver o que foi emprestado! Em C, utilize free. Em C++, delete. Em Java você pode esquecer...

FIM