BCSE102L- Structured and object-oriented programming

Dr. P.Keerthika
Associate Professor
School of Computer Science & Engineering
VIT, Vellore.

Dynamic Memory Allocation – C++

new and delete

Overloading of new & delete operators

Dynamic Memory Allocation

- **Functions**
  - **malloc()**
  - **calloc()**
  - **realloc()**
  - **free()**
- **Operators**
  - **new**
  - **delete**

new Operator

- Request for memory allocation on the free store.
- If sufficient memory is available, a new operator initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable.
- Syntax: pointer-variable = new data-type;
- Example: int *p = new int;
- Initialize using new:

```
int* p = new int(25);
float* q = new float(75.25);
```

- Allocate Block of Memory:

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

delete Operator

- to deallocate dynamically allocated memory
- Syntax:

```
delete pointer-variable;
```

- Example:

```
delete p;

delete[] p;
```

Dynamic Constructors

```
A()    //Default constructor
{
        value = new int;     //Memory allocation at run time
        *value = 1234;
}
A(int p_value)          //Parameterised constructor
{
        value = new int;        //Memory allocation at run time
        *value= p_value+3;
}
~A()
{
        delete value ;
}
};
```

Overloading of New & delete Operators

```
class A
{     int x;
public:
   void * operator new(size_t);
   void operator delete (void *);
};
void *A:: operator new(size_t s)
{
   void *ptr;
   ptr = malloc(s);
   return ptr;
}
void A::operator delete(void *s)
{
   free(s);
}
int main()
{
   A *a=new A;       //allocating for object
   cout<<"Allocated address:"<<a<<endl;
   delete a;
   cout<<"Deallocated!!!!"<<endl;
}
```