In programming there may be scenarios where programmers may not know the memory needed until run time. In this case, the programmer can opt to reserve as much memory as possible, assigning the maximum memory space needed to tackle this situation. This would result in wastage of unused memory spaces. Memory management operators are used to handle this situation in C++ programming language.

**What are memory management operators?**

There are two types of memory management operators in C++:

1. new
2. delete

These two memory management operators are used for allocating and freeing memory blocks in efficient and convenient ways.

**New operator:**

The new operator in C++ is used for dynamic storage allocation. This operator can be used to create object of any type.

**General syntax of new operator in C++:**

The general syntax of new operator in C++ is as follows:

pointer variable = new datatype;

In the above statement, new is a keyword and the pointer variable is a variable of type datatype.

**For example:**

int \*a=new int;

In the above example, the new operator allocates sufficient memory to hold the object of datatype int and returns a pointer to its starting point. The pointer variable a holds the address of memory space allocated.

Dynamic variables are never initialized by the compiler. Therefore, the programmer should make it a practice to first assign them a value.

The assignment can be made in either of the two ways:

Sample Code

1. int \*a = new int;
2. \*a = 20;

Copyright exforsys.com

or

Sample Code

1. int \*a = new int(20);

Copyright exforsys.com

**delete operator:**

The delete operator in C++ is used for releasing memory space when the object is no longer needed. Once a new operator is used, it is efficient to use the corresponding delete operator for release of memory.

**General syntax of delete operator in C++:**

The general syntax of delete operator in C++ is as follows:

delete pointer variable;

In the above example, delete is a keyword and the pointer variable is the pointer that points to the objects already created in the new operator. Some of the important points the programmer must note while using memory management operators are described below:

1. The programmer must take care not to free or delete a pointer variable that has already been deleted.
2. Overloading of new and delete operator is possible (to be discussed in detail in later section on overloading).
3. We know that sizeof operator is used for computing the size of the object. Using memory management operator, the size of the object is automatically computed.
4. The programmer must take care not to free or delete pointer variables that have not been allocated using a new operator.
5. Null pointer is returned by the new operator when there is insufficient memory available for allocation.

**Example:**

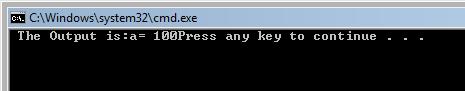
To understand the concept of new and delete memory management operator in C++:

Sample Code

1. #include <iostream>
2. using namespace std;
3. void main()
4. {
5. *//Allocates using new operator memory space in memory for storing a integer datatype*
6. int \*a= new int;
7. \*a=100;
8. [cout](http://www.opengroup.org/onlinepubs/009695399/functions/cout.html) << " The Output is:a= " << \*a;
9. *//Memory Released using delete operator*
10. delete a;
12. }

Copyright exforsys.com

The output of the above program is



In the above program, the statement:

int \*a= new a;

Holds memory space in memory for storing a integer datatype. The statement:

\*a=100

This denotes that the value present in address location pointed by the pointer variable a is 100 and this value of a is printed in the output statement giving the output shown in the example above. The memory allocated by the new operator for storing the integer variable pointed by a is released using the delete operator as: delete a;