Tutorial Link https://codequotient.com/tutorials/Dynamic memory allocation/5b3908e9c6a1d0259e728f8f

TUTORIAL

Dynamic memory allocation

Chapter

In dynamic allocation, the programmers handle the memory allocation and deallocation himself via new and delete operators. The new operator allocates the memory and returns a pointer to the start of it. The delete operator frees memory previously allocated using **new** operator. The general forms of new and delete are:

```
pointer_var = new type;
Delete pointer_var;
```

For example:

C++

```
#include<iostream>

using namespace std;

int main( )
{
  int *p;
  p=new int; // dynamically allocate an int and assign its address to  pointer
  *p=123;
  cout<< "At address " << p << " the value is "<< *p;
  delete p;
  return 0;
}


```

The memory can be allocated dynamically to arrays using ‘new’ and deallocated using ‘delete’ operators. The general form is

```
Pointer_var= new array_type[size];
Delete [ ] pointer_var;
```

An object can be created at runtime. Such an object is called dynamic object. The dynamically object acts just like any other object. When it is created, its constructor is called. When the object is freed, its destructor is executed. The general form is

```
Ptr = new < classname>;
delete ptr;
```

Let us take our example of Circle.

C++

```cpp
#include<iostream>

using namespace std;

class Circle
{
  private:
  double radius;
  double area;

  public:
  Circle() // Constructor
  {
    cout<<"\n constructor";
    area = 0;
    radius = 5.6;
  }
  ~Circle() // Destructor
  {
    cout<< "\n destructor";
  }
  void showArea( ) // Member function
  {
    area = radius * radius * 3.1416;
    cout<<"\nArea = "<< area;
  }
};

int main()
{
  Circle *C;
  C= new Circle;
  C -> showArea();
  delete C;
  return 0;
}



```

OUTPUT:

```
constructor
Area = 98.520576
destructor
```