अध्याय –10

कंस्ट्रक्टर और डिस्ट्रक्टर

10.1 परिचय

पिछले अध्याय में क्लासों के उदाहरणों में हम मेंम्बर फंक्शन जैसे input() का उपयोग किया है। क्लास के प्राईवेट डेटा मेंम्बर को इनिशियलाइज(प्रांरभिक वेल्यू देना) करने के लिए फंक्शन कॉल स्टेटमेंट का पहले से घोषित ऑब्जेक्ट के साथ प्रयोग किया जाता है। ये फंक्शन डेटा मेंम्बर को उनको ऑब्जेक्ट बनने के समय इनिशियलाइज करने में असमर्थ है। C++ का उद्धेश्य यह है कि क्लास भी बेसिक डेटा टाईप की तरह हो। जब एक क्लास टाईप वेरिएबल (ऑब्जेक्ट) को घोषित किये जाते है तब ठीक उसी तरह इनिशियलाइज हो जिस तरह से बेसिक डेटा टाईप के वेरिएबल को इनिशियलाइज किया जाता है।

इस अध्याय में, हम एक विशेष मेंम्बर फंक्शन जिसे कंस्ट्रक्टर कहा जाता है उसकी चर्चा करेगें। जो कि ऑब्जेक्ट को इनिशियलाइज करने के लिए काम आते है। जब इन्हें घोषित किया जाता है। एक दूसरा विशेष मेंम्बर फंक्शन डिस्ट्रक्टर जो उन ऑब्जेक्ट को खत्म करता है जो आगे के लिए काम नहीं आते है।

10.2 कंस्ट्रक्टर

कंस्ट्रक्टर एक विशेष प्रकार का मेंम्बर फंक्शन है उसके क्लास के ऑब्जेक्ट को इनिशियलाइज करने के लिए काम आते हैं। कंस्ट्रक्टर स्वतः ही कॉल हो जाते है जब उसके क्लास के ऑब्जेक्ट घोषित किया जाते है। कंस्ट्रक्टर फंक्शन की निम्नलिखित विशेषताएँ होती है

- इसका नाम क्लास के नाम जैसा ही होता है।
- ये क्लास के पब्लिक अनुभाग में घोषित होने चाहिए ।
- जब ऑब्जेक्ट घोषित किया जाते है तब ये स्वतः ही कॉल हो जाते है।
- इनका कोई रिटर्न टाईप नहीं होता है, यहाँ तक की void भी और इस प्रकार ये कोई वेल्यू रिटर्न नहीं करता है।
- इनको इनहेरिट नहीं किया जा सकता है।
- इनके एड्रेस को एक्सेस नहीं कर सकते है।

उदाहरण के लिए :–

```
class point
{
```

```
        int x,y;
public:
point(void);  //constructor declared
- - - - - - - - -
- - - - - - - - -
};
point :: point(void)  //constructor defined
{
        x=0;
        y=0;
}
```

जब हम क्लास point के ऑब्जेक्ट को घोषित करते हैं

उदाहरण के लिए :–

```
            point p;
```

क्लास के अन्दर कंस्ट्रक्टर स्वतः ही कॉल हो जाता है और प्राईवेट डेटा मेंम्बर x और y को शून्य से ऑब्जेक्ट p के लिए इनिशियलाइज कर देते है। जो कंस्ट्रक्टर आरग्यूमेन्ट नहीं लेते है उसे डिफॉल्ट कंस्ट्रक्टर कहा जाता है। अगर ऐसा कोई कंस्ट्रक्टर क्लास में परिभाषित नहीं है तब कम्पाइलर डिफॉल्ट कंस्ट्रक्टर उपलब्ध करवाता है उस क्लास के ऑब्जेक्ट घोषित करने के लिए।

10.3 पैरामीटरराइज्ड कंस्ट्रक्टर

जो कंस्ट्रक्टर आरग्यूमेन्ट लेते है उन्है पैरामीटराइज्ड कंस्ट्रक्टर कहा जाता है।

उदाहरण के लिए :–

```
class point
{
        int x, y;
public:
        point(int a, int b);    // parameterized constructor
        {
        - - - - - - - -
```

```
        --------
        }
};
point::point(int a, int b)
{
        x=a;
        y=b;
}
```

पैरामीटराइज्ड कंस्ट्रक्टर को दो तरीको से कॉल किया जा सकता है।

- point p= point(10,20);   //explicit call

यह स्टेटमेंट एक ऑब्जेक्ट pको 10 और 20 वेल्यू से इनिशियलाइज करता हैं ।

- point p(10,20);   //implicit call

यह स्टेटमेंट उपरोक्त स्टेटमेंट की तरह काम करता है।

कंस्ट्रक्टर फंक्शन को इनलाईन फंक्शन के रूप में भी परिभाषित किया जा सकता है।

उदाहरण के लिए :–

```
class point
{
        int x, y;
public:
      point(int a, int b)
      {
            x=a;
            y=b;
      }
--------
--------
};
```

कंस्ट्रक्टर के ऑरग्यूमेन्ट किसी भी टाईप के हो सकते है सिवाय उस क्लास के जिससे ये सम्बंधित हैं।

उदाहरण के लिए :—

```
class X
{
--------
--------
public:
X(X);
};
```

अमान्य है।

लेकिन एक कंस्ट्रक्टर अपनी क्लास का रेफरेंस एक ऑरग्यूमेन्ट के रूप में ले सकता है।

उदाहरण के लिए :—

```
class X
{
--------
--------
public:
X(X&);
};
```

यह वैद्य है और कंस्ट्रक्टर को कॉपी कंस्ट्रक्टर कहा जाता है।

प्रोग्राम 10.1: पैरामीटराइज्ड कंस्ट्रक्टर

```
#include<iostream>
using namespace std;
class rectangle
{
        int length;
        int breadth;
public:
        rectangle(int a, int b)
```

```
        {
                length=a;
                breadth=b;
        }
        void area()
        {
                cout<<"Area="<<length*breadth;
        }
};
int main()
{
        rectangle r(5,10);
        return 0;
}
```

प्रोग्राम 10.1 का आउटपुट होगा–

Area=50

10.4 कंस्ट्रक्टर ओवरलोडिंग :–

एक क्लास में एक से ज्यादा कंस्ट्रक्टर हो सकते है और इसे कंस्ट्रक्टर ओवरलोडिंग कहा जाता हैं ।

प्रोग्राम 10.2: कंस्ट्रक्टर ओवरलोडिंग

```
#include<iostream>
using namespace std;
class point
{
         int x,y;
public:
        point()// no argument constructor
        {
```

```
		x=0;
		y=0;
	}
	point(int a)    //one argument constructor
	{ x=y=a;}
	point(int m, int n)    //two arguments constructor
	{
		x=m;
		y=n;
	}
	void show()
	{
		cout<<“x=”<<x<<“\n”;
		cout<<“y=”<<y<<“\n”;
	}
};
int main()
{
	point p1;
	point p2(5);
	point p3(7,11);
	cout<<“Coordinates of p1 are\n”;
	p1.show();
	cout<<“Coordinates of p2 are\n”;
	p2.show();
	cout<<“Coordinates of p3 are\n”;
	p3.show();
	return 0;
}
```

प्रोग्राम 10.2 का आउटपुट होगा–

Coordinates of p1 are

x=0

y=0

Coordinates of p2 are

x=5

y=5

Coordinates of p3 are

x=7

y=11

उपरोक्त प्रोग्राम में, क्लास point में तीन कंस्ट्रक्टर है। पहला बिना आरग्यूमेन्ट का कंस्ट्रक्टर है। इसका उपयोग बिना किसी प्रारंभिक वेल्यू का ऑब्जेक्ट बनाने के लिए किया जाता है। जब हम अन्य कंस्ट्रक्टर परिभाषित करते हैं तब हमें कम्पाइलर को संतुष्ट करने के लिए बिना आऱग्यूमेन्ट का कंस्ट्रक्टर परिभाषित करना होगा। दूसरा कंस्ट्रक्टर एक वेल्यू पैरामीटर के रूप में लेता है और ऑब्जेक्ट को इनिशियलाइज करता है। तीसरा कंस्ट्रक्टर दो आऱग्यूमेन्ट लेता है और इन दो वेल्यू से ऑब्जेक्ट को इनिशियलाइज करता है।

10.5 डिफॉल्ट आऱग्यूमेन्ट के साथ कंस्ट्रक्टर

कंस्ट्रक्टर डिफॉल्ट आऱग्यूमेन्ट ले सकते है।

उदाहरण के लिए :–

```
point(int a,  int b=0);
```

ध्यान दे कि डिफॉल्ट आऱग्यूमेन्ट दायें से बायें दिये जाते है। आऱग्यूमेन्ट bकी डिफॉल्ट वेल्यू शून्य है। तब यह स्टेटमेंट

```
point p(5);
```

aको 5 वेल्यू असाइन करता है और bको शून्य (डिफॉल्ट द्वारा) लेकिन स्टेटमेंट

```
point(7,11);
```

a को 7 वेल्यू असाइन करता है और b को 11 क्योंकि जब वास्तविक पैरामीटर दिये गये हो तब वे डिफॉल्ट आरग्यूमेन्ट को ओवरराइड कर देते है।

अगर एक आरग्यूमेन्ट वाला कंस्ट्रक्टर भी इस कंस्ट्रक्टर के साथ मौजूद है तब कॉल करने वाला स्टेटमेंट

```
point p(5);
```

यह फैसला लेने में असमर्थ है कि कौनसे कंस्ट्रक्टर को कॉल किया जाये और एक अस्पष्टता की स्थिति उत्पन्न हो जाती है। कम्पाइलर एक त्रुटि संदेश उत्पन्न कर देता है।

10.6 ऑब्जेक्ट का डाइनामिक इनिशियलाइजेशन

एक ऑब्जेक्ट की इनिशियल वेल्यू रन टाइम पर दी जा सकती है। डाइनामिक इनिशियलाइजेशन का फायदा यह है कि हम विभिन्न प्रकार के इनपुट कंस्ट्रक्टर ओवरलोडिंग के द्वारा दे सकते है।

प्रोग्राम 10.3 ऑब्जेक्ट का डाइनामिक इनिशियलाइजेशन

```
#include<iostream>
using namespace std;
class shape
{
        float length, breadth;
        float radius;
        float area;
public:
        shape() { }
        shape(float r)
        {
                radius=r;
                area=3.14*r*r;
        }
        shape(float l, float b)
        {
                length=l;
                breadth=b;
                area=length*breadth;
        }
        void display()
        {
                cout<<"Area="<<area<<"\n";
        }
};
int main()
{
```

```
        shape circle, rectangle;
        float r, l, b;
        cout<<"Enter the radius of circle\n";
        cin>>r;
        circle=shape(r);
        cout<<"Enter the length and breadth of rectangle\n";
        cin>>l>>b;
        rectangle=shape(l,b);
        cout<<"Area of circle\n";
        circle.display();
        cout<<"Area of rectangle\n";
        rectangle.display();
        return 0;
}
```

प्रोग्राम 10.3 का आउटपुट होगा–

```
Enter the radius of circle
5
Enter the length and breadth of rectangle
17      8
Area of circle
78.5
Area of rectangle
136
```

10.7 कॉपी कंस्ट्रक्टर

कंस्ट्रक्टर जो एक ऑब्जेक्ट को उसी क्लास के दूसरे ऑब्जेक्ट से घोषित और इनिशियलाइज करता है, उसे कॉपी कंस्ट्रक्टर कहा जाता है। एक कॉपी कंस्ट्रक्टर उसी क्लास के ऑब्जेक्ट का रेफरेंस आरग्यूमेन्ट के रूप में लेते है।

प्रोग्राम 10.4 कॉपी कंस्ट्रक्टर

```
#include<iostream>
using namespace std;
class product
{
        int code;
```

```
public:
        product(){  } // default constructor
        product(int x)//parameterized constructor
        {
                code=x;
        }
        product(product &y)          //copy constructor
        {
                code=y.code;          //copy the value
        }
        void display(void)
        {
                cout<<code;
        }
};
int main()
{
        product p1(10);
        product p2(p1);       //copy constructor called
        product p3=p1;        //again copy constructor called
        cout<<"Code of p1:";
        p1.display();
        cout<<"\nCode of p2:";
         p2.display();
        cout<<"\nCode of p3:";
        p3.display();
        return 0;
}
```

प्रोग्राम 10.4 का आउटपुट होगा–

Code of p1:10

Code of p2:10

Code of p3:10

नोटः— जब प्रोग्राम में कॉपी कंस्ट्रक्टर परिभाषित नहीं होता है तब कम्पाइलर अपना कॉपी कंस्ट्रक्टर उपलब्ध कर देता है।

10.8 डिस्ट्रक्टर :—

क्लास का एक विशेष प्रकार का मेंम्बर फंक्शन जिसका प्रयोग कंस्ट्रक्टर के द्वारा बनाये गये ऑब्जेक्ट को खत्म करने के लिए किया जाता है। डिस्ट्रक्टर फंक्शन की निम्नलिखित विशेषताएँ होती है

(1) इसका नाम क्लास के नाम के जैसा होता है। लेकिन इसके नाम से पहले tilde(~) चिन्ह होता है।

(2) यह आरग्यूमेन्ट नहीं लेता है और वेल्यू भी रिटर्न नहीं करता है।

(3) जब किसी प्रोग्राम या ब्लोक या फंक्शन से बाहर आते है तब यह कम्पाइलर के द्वारा स्वतः ही कॉल होता है

निम्नलिखित प्रोग्राम दर्शाता है कि डिस्ट्रक्टर फंक्शन कम्पाइलर के द्वारा स्वतः ही कॉल होता है।

प्रोग्राम 10.5 डिस्ट्रक्टर फंक्शन

```
#include<iostram>
using namespace std;
class sample
{
sample()                    // Constructor
{
cout<<“Object created\n”;
}
~sample()                   // Destructor
{
cout<<“Object destroyed”;
}
};

int main()
{
```

```
sample s;
 return 0;
}
```

प्रोग्राम 10.5 का आउटपुट होगा–

```
Object created
Object destroyed
```

डिस्ट्रक्टर का उपयोग ऑब्जेक्ट को आंवटित मेमौरी को रन टाइम पर मुक्त करने के लिए किया जाता है। ताकि मुक्त मेमौरी का पुनः उपयोग दूसरे प्रोग्राम या ऑब्जेक्ट के लिए किया जा सके। किसी ऑब्जेक्ट को मेमौरी का आवंटन कंस्ट्रक्टर फंक्शन में new ऑपरेटर से किया जाता है और डिस्ट्रक्टर फंक्शन में delete ऑपरेटर से आवंटित मेमौरी को पुनः लिया जाता है।

प्रोग्राम 10.6 डिस्ट्रक्टर का उपयोग मेमौरी को मुक्त करने के लिए

```
#include<iostream>
using namespace std;
class sample
{
        char *t;
public:
        sample(int length)
        {
        t=new char[length];
        cout<<"Character array of length"<<length<<"created";
        }
        ~sample()
        {
        delete t;
        cout<<"\nMemory de-allocated for the character array";
        }
};
int main()
{
```

```
sample s(10);
return 0;
}
```

प्रोग्राम 10.5 का आउटपुट होगा–

Character array of length 10 created

Memory de-allocated for the character array

महत्वपूर्ण बिंदु

- कंस्ट्रक्टर एक विशेष प्रकार का मेंम्बर फंक्शन है यह उसके क्लास के ऑब्जेक्ट को इनिशियलाइज करने के लिए काम आते हैं।
- कंस्ट्रक्टर के ऑरग्यूमेन्ट किसी भी टाईप के हो सकते है सिवाय उस क्लास के जिससे ये सम्बंधित हैं।
- एक कंस्ट्रक्टर अपनी क्लास का रेफरेंस एक ऑरग्यूमेन्ट के रूप में ले सकता है।
- कंस्ट्रक्टर जो एक ऑब्जेक्ट को उसी क्लास के दूसरे ऑब्जेक्ट से घोषित और इनिशियलाइज करता है, उसे कॉपी कंस्ट्रक्टर कहा जाता है।
- जब प्रोग्राम में कॉपी कंस्ट्रक्टर परिभाषित नहीं होता है तब कम्पाइलर अपना कॉपी कंस्ट्रक्टर उपलब्ध कर देता है।
- क्लास का एक विशेष प्रकार का मेंम्बर फंक्शन जिसका प्रयोग कंस्ट्रक्टर के द्वारा बनाये गये ऑब्जेक्ट को खत्म करने के लिए किया जाता है उसे डिस्ट्रक्टर कहा जाता है।

अभ्यासार्थ प्रश्न

वस्तुनिष्ठ प्रश्न :

प्रश्न 1.इनमें से कोनसाकंस्ट्रक्टरस के संदर्भ में सत्य है?

(अ) इसका नाम क्लास के नाम जैसा ही होता है।

(ब) ये क्लास के पब्लिक अनुभाग में घोषित होने चाहिए ।

(स) जब ऑब्जेक्ट घोषित किये जाते है तब ये स्वतः ही कॉल हो जाते है।

(द) उपरोक्त सभी

प्रश्न 2. कंस्ट्रक्टर जो आरग्यूमेन्ट लेते है उन्हेंक्या कहा जाता है?

(अ) डिफॉल्ट कंस्ट्रक्टर (ब) बिना आऱग्यूमेन्ट का

(स) पैरामीटराइज्ड कंस्ट्रक्टर (द) इनमें से कोई नहीं

प्रश्न 3. कंस्ट्रक्टर जो एक ऑब्जेक्ट को उसी क्लास के दूसरे ऑब्जेक्ट से घोषित और इनिशियलाइज करता है, उसे कहा जाता है।

(अ) डिफॉल्ट कंस्ट्रक्टर (ब) कॉपी कंस्ट्रक्टर

(स) पैरामीटराइज्ड कंस्ट्रक्टर (द) इनमें से कोई नहीं

प्रश्न 4. इनमें से कोनसाडिस्ट्रक्टरस के संदर्भ में सत्य है?

(अ) इसका नाम क्लास के नाम के जैसा होता है। लेकिन इसके नाम से पहले tilde(~) चिन्ह होता है।

(ब) यह आरग्यूमेन्ट नहीं लेता है और वेल्यू भी रिटर्न नहीं करता है।

(स) यह कम्पाइलर के द्वारा स्वतः ही कॉल होता है जब किसी प्रोग्राम या ब्लोक या फंक्शन से बाहर आते है।

(द) उपरोक्त सभी

अति लघूउत्तरात्मक प्रश्न

प्रश्न 1. कंस्ट्रक्टरस क्या होते हैं ?

प्रश्न 2. पैरामीटराइज्ड कंस्ट्रक्टरस क्या होते हैं ?

प्रश्न 3. कंस्ट्रक्टर ओवरलोड़िग किसे कहते है ?

लघूउत्तरात्मक प्रश्न

प्रश्न 1. कंस्ट्रक्टरस की क्या विशेषताएँ होती हैं ?

प्रश्न 2. डिस्ट्रक्टरस क्या होते है इसकी विशेषताएँ लिखों ?

प्रश्न 3. डिस्ट्रक्टरस के उपयोग लिखों ?

निबंधात्मक प्रश्न

प्रश्न 1. डिफॉल्ट आरग्यूमेन्ट के साथ कंस्ट्रक्टर का वर्णन कीजिए ?

उत्तरमाला

1:द 2:स 3:ब 4: द