

object-oriented programming (OOP)

KIAN_ACADEMY

Destructor :-

special methods in class Why ?

- ▶ 1 - name is the same name of the class
- ▶ 2 - has not a return type
- ▶ 3 - must not return any values



Note :-

- 1 - The Destructor is called automatically when an object life time is ended.
- 2 - Destructor are typically public

Why We are using constructor ?

- 1 - To destroy the object from memory (delete).
- 2 - to deallocate memory that was allocated for the object by the constructor

► Note :-

the Destructor destroy the object from down to top.

Example :-

public :

~Rectangle()// tilde (~)

{

cout<<"The program ended"<<endl;

}



2-Note :-

- 1 - If there is no Destructor in a class, compiler automatically creates a default Destructor
- 2- Cannot be declared as const, volatile, or static.

Important note:

There has to be only one Destructor in a class(**default Destructor**).

Quick review on Pointer:

```
int main()
{
    int *p;
    int x=25;
    p=&x;
    float *l;
    float y=45;
    l=&y;
    cout<<*p<<endl;
    cout<<l<<endl;
```

```
    cout<<&x<<endl;
    cout<<&p<<endl;
    cout<<*l<<endl;
}
```

The output is :

```
1 - 25
2 - 0x62fe08
3 - 0x62fe0c
4 - 0x62fe10
5 - 45
```

New And Delete :-

► New Operator

The new operator denotes a request for memory allocation on the Free Store .

If sufficient memory is available, new operator initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable.

Syntax to use new operator :-

pointer-variable = new data-type ;

Example :-

```
int *p;
```

```
p = new int;
```

```
*p=18;
```

or

```
int *p = new int;
```

```
*p=18;
```

or

```
int *p = new int(18);
```


New And Delete :-

► delete operator

Since it is programmer's responsibility to deallocate dynamically allocated memory, programmers are provided delete operator by C++ language.

Syntax to use delete operator :-

delete pointer-variable;

Example :-

```
int main()
{
    int *p;
    p=new int;
    *p=10;
    cout<<*p<<endl;
    delete p;
    cout<<*p<<endl;
}
```

The output is :

1-10

2-16848880



Note :-

We are using New in Constructor And delete in Destructor....HOW ?

Example :-

class Rectangle//.h

```
{
private:
int *width,*height;
public:
Rectangle(int w, int h );
~Rectangle();
int area()
{
return(*width * *height);
}
}
```

**class
Rectangle//.cpp**

```
{
Rectangle::Rectangle(int w, int h )
{
/*
width=w;//error
height=h; //error
*/
}
```

```
width=new int;
height=new int;
*width=w;
*height=h;
}
Rectangle::~~Rectangle()
{
delete width;
delete height;
}
}
```

int main

```
{
Rectangle r1 (4,5);
Rectangle r2(7,8);
cout<<r1.area()<<endl;
cout<<r2.area()<<endl;
}
```

The output is :

1-20

2-56

Copy Constructor

Copy Constructors :- is a type of constructor which is used to create a copy of an already existing object of a class type.

- It is another way to initialize an object:
- Used to initialize an object with another object of the same type.

Example :-

```
#include<iostream>
using namespace std;
class copyconstructor
{
private:
    int x, y; //data members
public:
    copyconstructor(int x1, int y1)
    {
        x = x1;
        y = y1;
    }
}
```

```
void display()
{
    cout<<"X is :"<<x<<"\t"<<"Y is :"<<y<<endl;
}
};

int main()
{
//*****
copyconstructor obj1(10, 15); //
Normal constructor
copyconstructor obj2 = obj1; //
Copy constructor
copyconstructor obj3(obj1); // Copy
constructor
//*****
}
```

```
cout<<"Normal constructor : "<<endl;;
obj1.display();
cout<<"Copy constructor : "<<endl;
obj2.display();
cout<<"Copy constructor : "<<endl;
obj3.display();
return 0;
}
```

The output is:

Normal constructor :

X is :10 Y is :15

Copy constructor :

X is :10 Y is :15

Copy constructor : X is :10 Y is :15

Exam questions:

- ▶ (1) It is a _____ error to pass arguments to a destructor.
- ▶ A - logical
- ▶ B - virtual
- ▶ C - syntax
- ▶ D - linker



▶ **(2)** Which of the following are NOT provided by the compiler by default?

▶ A - Zero-argument Constructor

▶ B - Destructor

▶ C - Copy Constructor

▶ D - Copy Destructor



▶ **(3)** Which of the following cannot be declared as virtual?

- ▶ A - Constructor
- ▶ B - Destructor
- ▶ C - Data Members
- ▶ D - Both A and C



▶ (4) Constructors _____ to allow different approaches of object construction.

▶ A - cannot overloaded

▶ B - can be overloaded

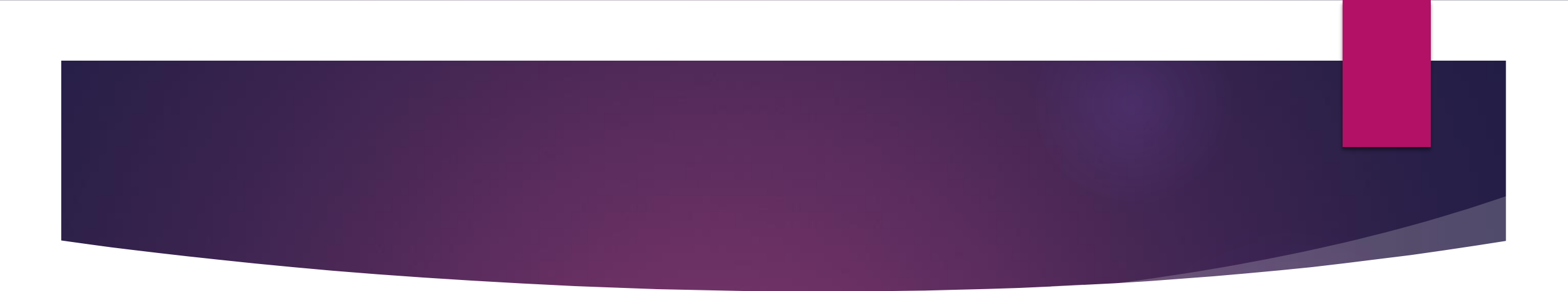
▶ C - can be called

▶ D - can be nested



▶ **(5)** Which of the following gets called when an object goes out of scope?

- ▶ A - constructor
- ▶ B - destructor
- ▶ C - main
- ▶ D - virtual function



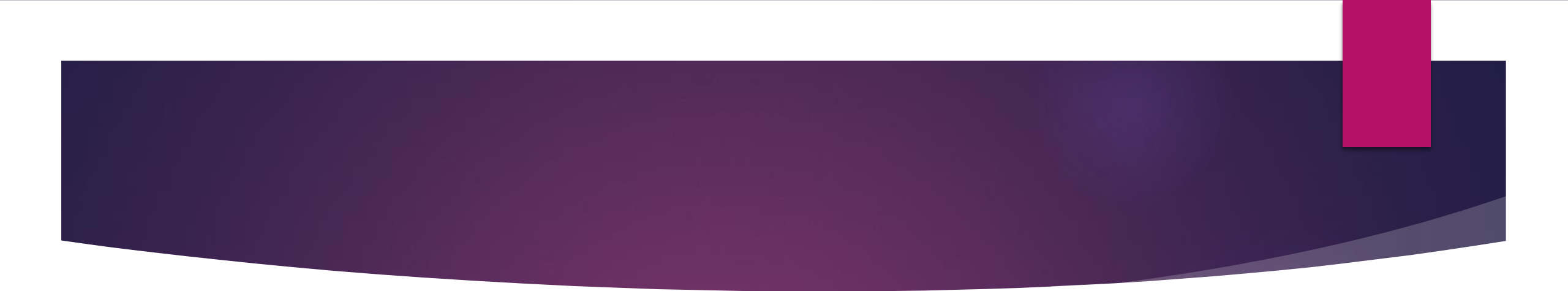
▶ **(6)** A union that has no constructor can be initialized with another union of _____ type

- ▶ A - different
- ▶ B - same
- ▶ C - virtual
- ▶ D - class



► **(7)** When are the Global objects destroyed?

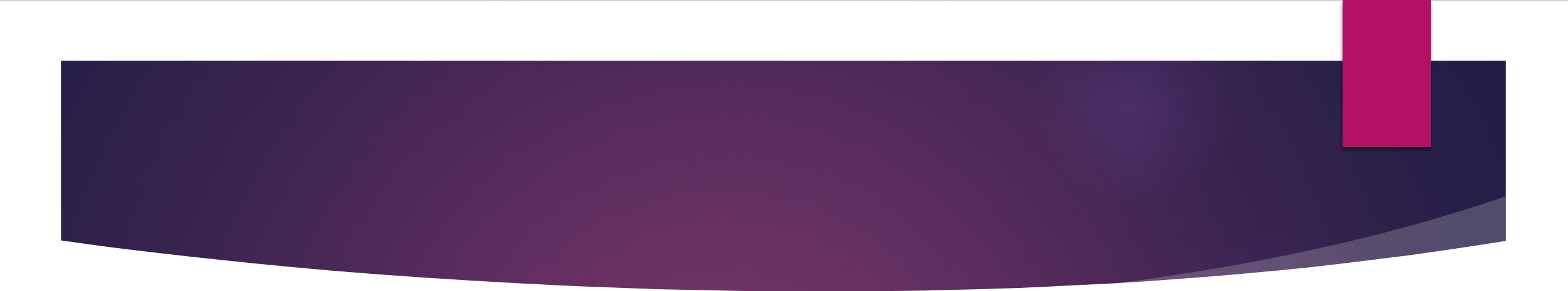
- A - When the control comes out of the block in which they are being used.
- B - When the program terminates.
- C - When the control comes out of the function in which they are being used.
- D - As soon as local objects die.

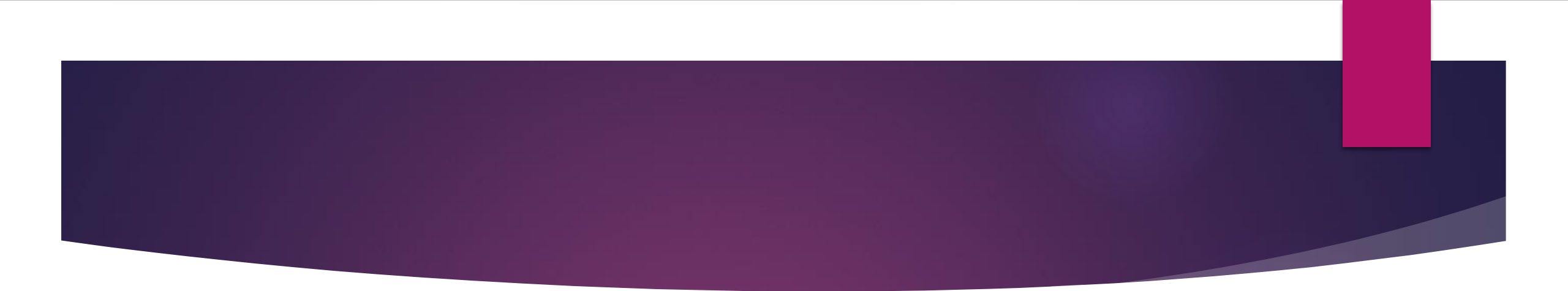
- 
- ▶ **(8)** Which of the following statement is incorrect ?
 - ▶ A - Constructor is a member function of the class.
 - ▶ B - The compiler always provides a zero argument constructor.
 - ▶ C - It is necessary that a constructor in a class should always be public.
 - ▶ D - Both B and C.



► **(9)** Which of the following statement is correct?

- A - Constructor has the same name as that of the class.
- B - Destructor has the same name as that of the class with a tilde symbol at the beginning.
- C - Both A and B.
- D - Destructor has the same name as the first member function of the class.

- 
- ▶ **(10)** Which constructor function is designed to copy objects of the same class type?
 - ▶ A - Create constructor
 - ▶ B - Object constructor
 - ▶ C - Dynamic constructor
 - ▶ D - Copy constructor

- 
- ▶ **(11)** For automatic objects, constructors and destructors are called each time the objects
 - ▶ A - enter and leave scope
 - ▶ B - inherit parent class
 - ▶ C - are constructed
 - ▶ D - are destroyed



▶ **(12)** Destructor has the same name as the constructor and it is preceded by

▶ A - !

▶ B - ?

▶ C - ~

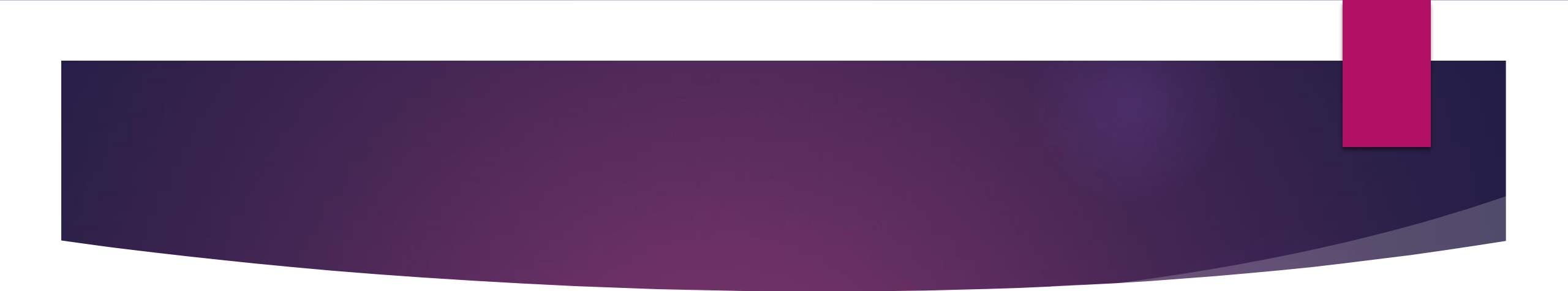
▶ D - \$



▶ **(13)** Can a class have virtual destructor ?

▶ A - yes

▶ B - no

- 
- ▶ **(14)** What happens when a class with parameterized constructors and having no default constructor is used in a program and we create an object that needs a zero-argument constructor?
 - ▶ A - Compile-time error.
 - ▶ B - Preprocessing error.
 - ▶ C - Runtime error.
 - ▶ D - Runtime exception.



▶ **(15)** A constructor that accepts _____ parameters is called the default constructor

▶ A - one

▶ B - two

▶ C - no

▶ D - three



▶ **(16)** How many parameters does a default constructor require?

▶ A - 1

▶ B - 2

▶ C - 0

▶ D - 3



▶ **(17)** How many types of constructors are there in C++?

▶ A - 1

▶ B - 2

▶ C - 3

▶ D - 4



▶ **(18)** What is the role of destructors in Classes ?

- ▶ A - To modify the data whenever required
- ▶ B - To destroy an object when the lifetime of an object ends
- ▶ C - To initialize the data members of an object when it is created
- ▶ D - To call private functions from the outer world

Answer the questions :-

Unfortunately, I forgot to put it 😁