C++ in depth

Pointers and DMA

Saurabh Shukla (MySirG)

Agenda

1. Object Pointer
2. this
3. DMA : new & delete

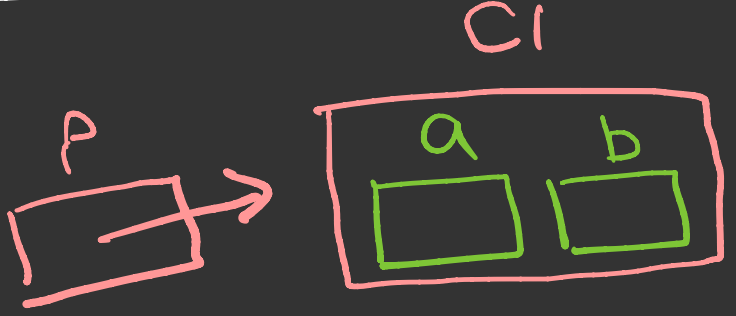
Pointers

- Pointer is a variable which contains address of another variable.
- Pointer is declared using * symbol
- Pointer content is considered address of the variable of type same as the type of the pointer
- Size of Pointer is fixed and depends on OS and Computer architecture

Object Pointer

```
Complex c1;
Complex *p;
```

*p ≅ c1




```
p = &c1;
```

c1.a c1.b

(*p).a (*p).b

p->a p->b

A pointer of class type which contains address of an object is object pointer.

this Pointer

- this is a keyword
- this is a local object pointer in every instance member function which contains address of the current object.
- this cannot be modified (you cannot change the value in this pointer)

When to use this pointer?

1. Name conflict between instance member variables and local variables

2. whenever it is required to represent current object in instance member function.

Types of variables in C++

Accordmy to scope

		life	scope
①	Global Variable	Program	Global
②	local Variable	function	function member funct.
③	instance variable	Object	Object . —
④	static member variable	Program	member fun class :: —
⑤	static local variable	Program	functin

DMA (V. Imp)

SMA (Stack)

Static Memory Allocation

```
int a;
float k;
char ch;
int *p;
struct Book b1;
 Book b2;
Complex c1;
Complex *ptr;
```

Compile Time

DMA (Heap)

Dynamic Memory Allocation

- malloc() — C/C++
- calloc() — C/C++
- free() — C/C++

- No name of DMA variables
- Life of DMA variable is
  - → Till the end of program, or
  - → until free() function explicitly release its memory.

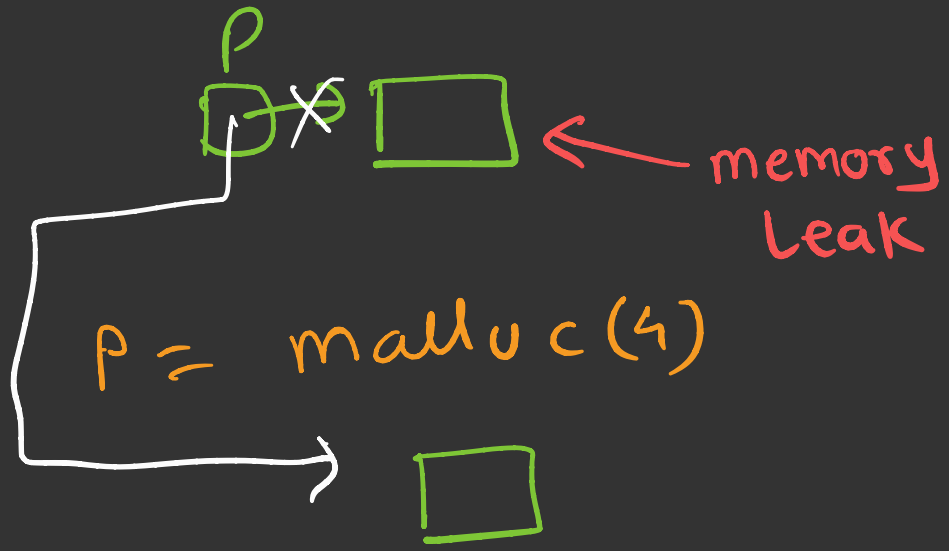
Recall

① Memory Leak

② Null Pointer

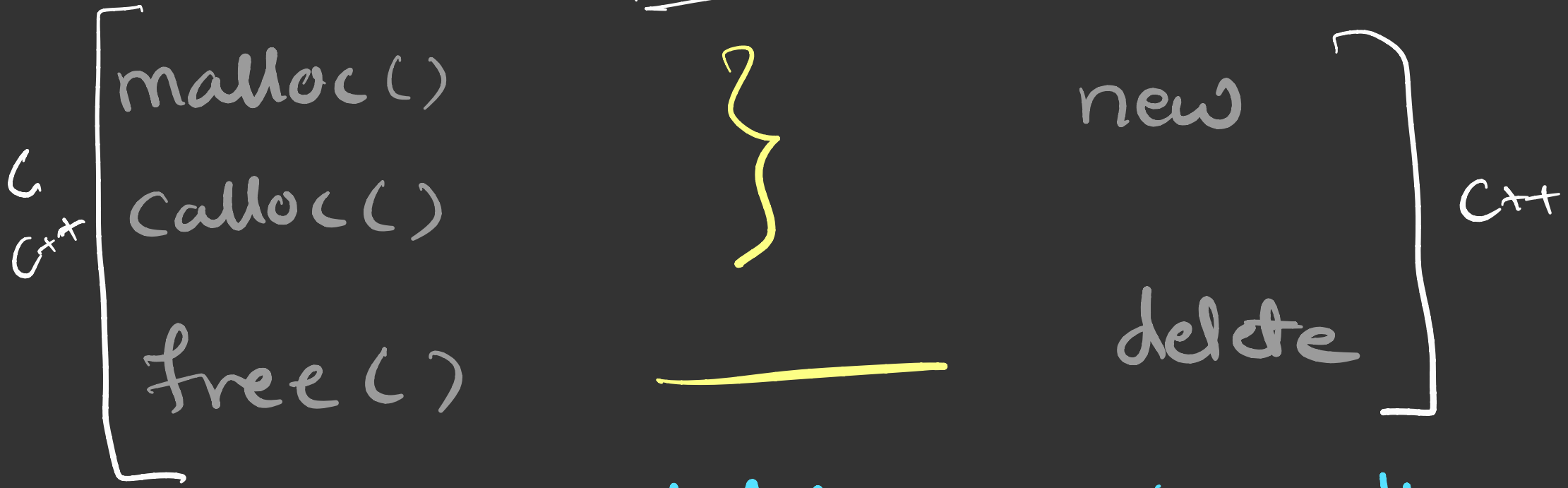
int *p;

p = malloc(4)



int *p; Wild pointer

int *p = NULL; NULL pointer



- new and delete are keywords
- new and delete are operators also.

```
ptr = new datatype;
```

new and delete

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


```
float *ptr = new float[5];
```


```
*(ptr+i)
ptr[i]
```

```
Complex *q = new Complex;
```


```
Complex *q = new Complex();
```

```
delete p;
delete q;
delete []ptr;
```