

C++ in depth

Pointers and DMA



Saurabh Shukla (MySirG)

Agenda

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

Pointers

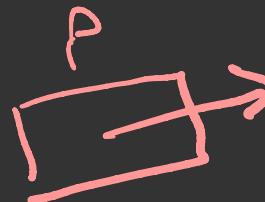
- Pointer is a variable which contains address of another variable.
- Pointer is declared using * symbol
- Pointer content is considered address of the variable of type same as the type of the pointer
- Size of Pointer is fixed and depends on OS and Computer architecture

Object Pointer

Complex C1;

Complex *P;

$*P \approx C1$



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++

According to scope

		<u>life</u>	<u>scope</u>
①	Global Variable	Program	Global
②	local variable	function	function
③	instance variable	Object	member func. Object - -
④	static member variable	Program	member fun class :: -
⑤	static local variable	Program	function

DMA (v. imp)

SMA (Stack)

Static Memory Allocation

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

Compile

Time

malloc()

calloc()

free()

No name of DMA variables

- Life of DMA variable is
 - Till the end of program, or
 - until free() function explicitly release its memory.

DMA (Heap)

Dynamic Memory Allocation

C/C++

C/C++

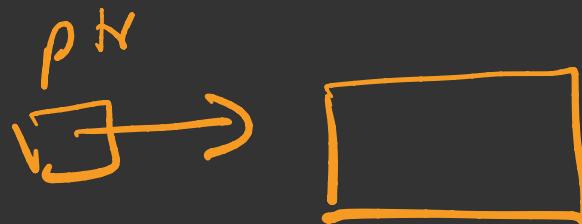
C/C++

Recall

① Memory Leak

② Null Pointer

DMA



`int *p = NULL;`



C++

malloc() → C/C++ new C++
calloc() → C/C++
free() → C/C++ delete C++

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

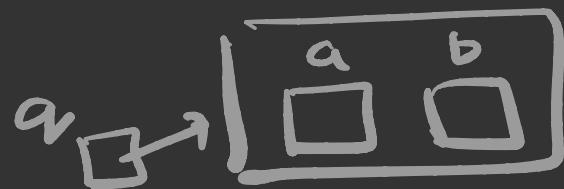
ptr = new datatype;

new and delete

int * P = new int ; float *ptr = new float [5];



Complex *q = new Complex;



Complex *q = new Complex();



delete P;
delete q;
delete []ptr;