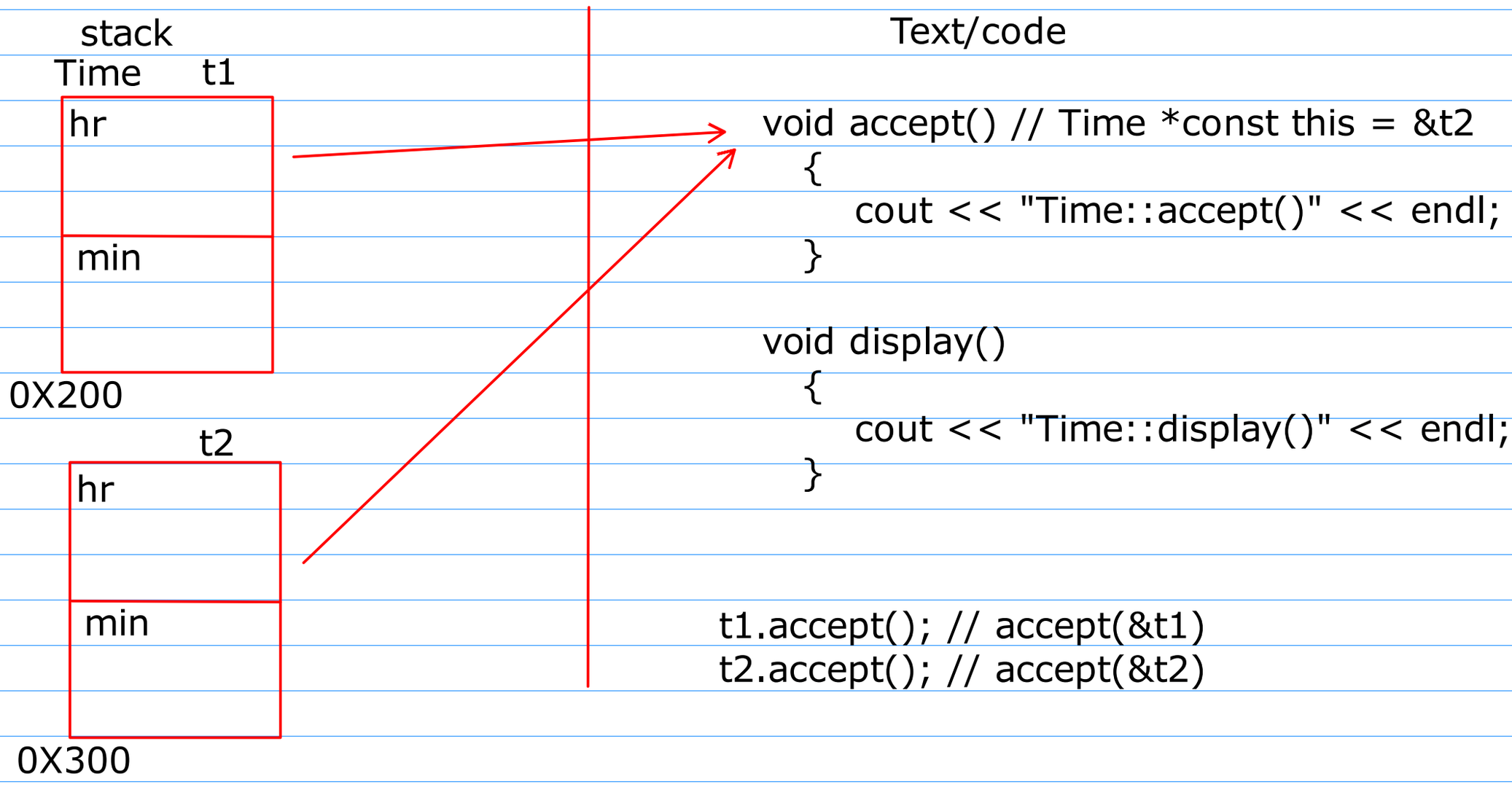
this pointer

- It is a pointer that is passed internally to all the non static member functions of the class
- It stores the address of the current calling object
- using this inside the function body is totally optional



Types of member functions

1. Constructor
2. Destructor
3. Mutator
4. Inspector
5. Facilitator

1. Constructor

- It is a special member function of a class
    1. The name of the ctor function is same as that of the class name
    2. Ctor does not have any return type
    3. It gets automatically called when object is created

#Types of Constructor

1. Defualt/Parameterless Ctor
2. Parameterized Ctor
3. Copy Ctor

```
Time(){
hr=10;
min=10;
}

Time(int,int){

}
```

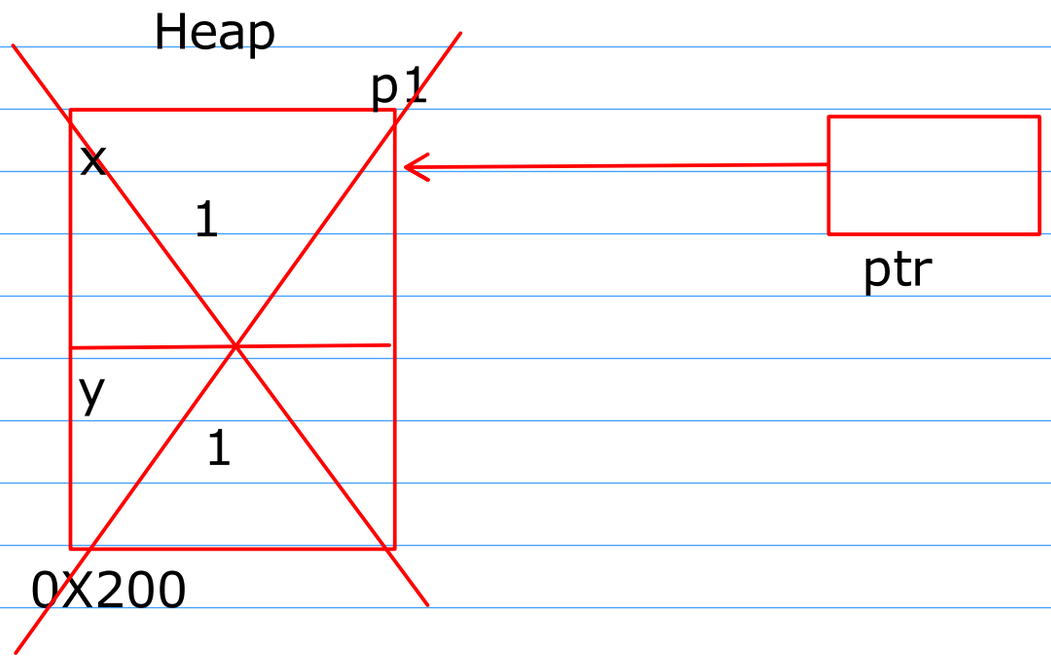
```
class Customer{
.
.
.
.
.
.
}
```

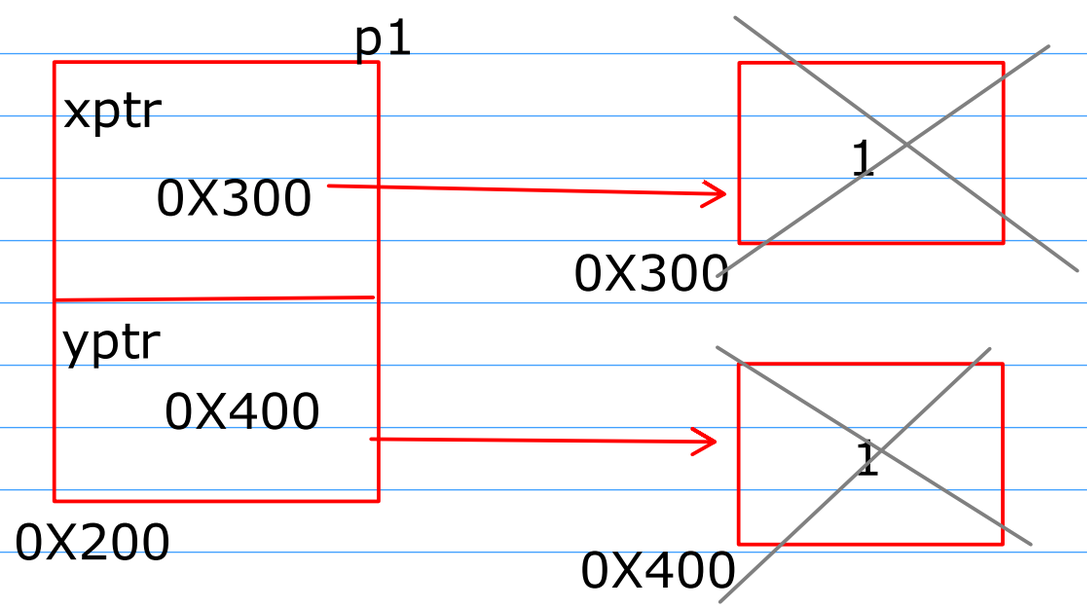
Ctor members initializer list

- It is a way of initializing the data members of the class in the same way as per their declaration order.
- It is also used for initializing the constant data members

Ctor Delegation

- It is a way of reusing the ctor body.
- one ctor can delegate the task of initialization to another ctor





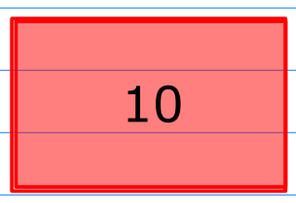
Destructor

- It is a special member function of a class
    1. The name of the dtor function is same as that of the class name with a tild(~)
    2. Dtor does not have any return type
    3. It gets automatically called when object goes out of scope

constant data memebers
constant memeber functions
constant object

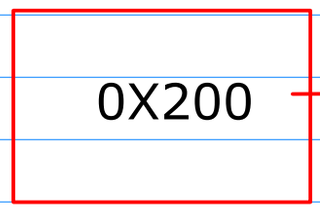
constant variable
constant pointer

const int num1 = 10

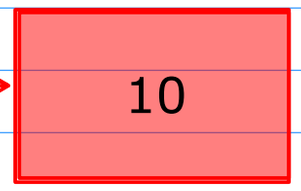


num1

num1 = 20; // NOT OK



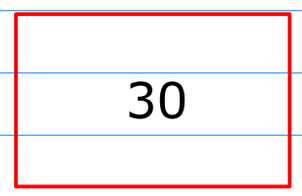
ptr



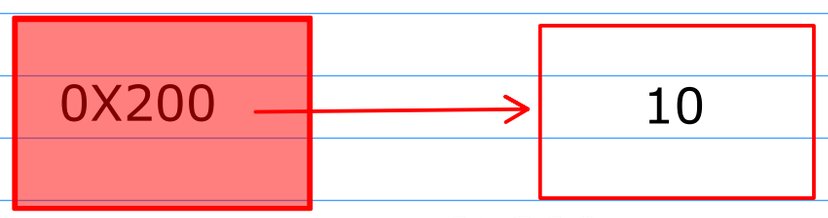
0X200 num1

const int num1 = 10;
const int *ptr = &num1;
//num1 = 20; NOT OK
//*ptr = 20;

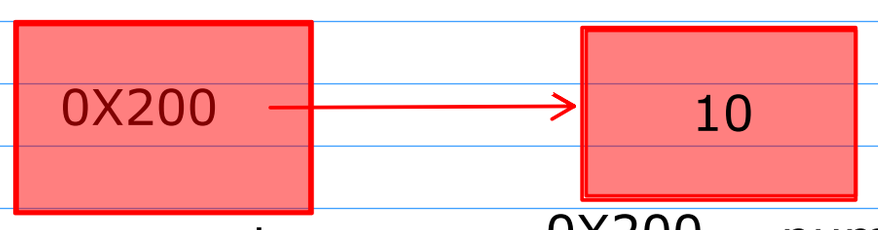
ptr = &num2//OK



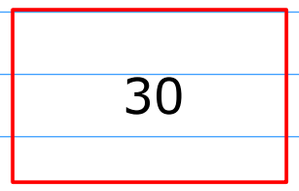
0X300 num2



ptr 0X200 num1



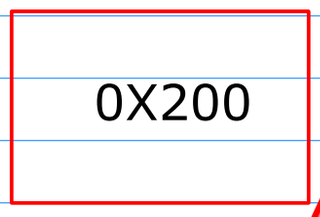
ptr 0X200 num1



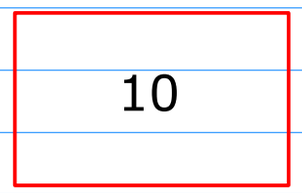
0X300 num2

int num1 = 10;
int *const ptr = &num1;
int num2 = 30;
//ptr = &num2; // NOT OK
*ptr = 20;

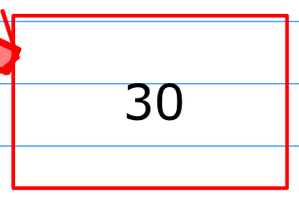
const int num1 = 10;
const int *const ptr = &num1;
num1 = 20;// NOT OK
*ptr = 20; // NOT OK
int num2 = 30;
ptr = &num2 ; // NOT OK



ptr



0X200 num1



num2

int num1 = 10;
const int *ptr = &num1;
num1 = 20;
int num2 = 30;
ptr = &num2;
//*ptr = 20; // NOT OK

1. Constant Data Member
- It must be initialized inside ctor members initializer list
- Once initialized we cannot change the value inside any functions

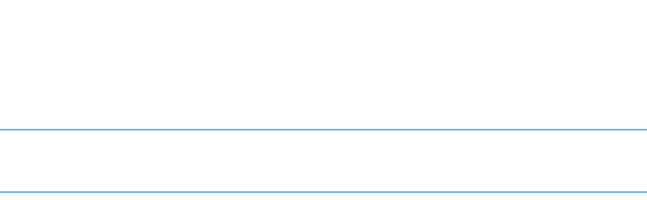
2. Constant Member function
    - It cannot change the state of an object inside it.
    - We cannot change the value inside the data members in these constant functions

3. Constant Object
    - Once created the state of the object cannot be changed.
    - we cannot call non constant member functions on these constant objects
    - we can call only constant member functions on constant objects

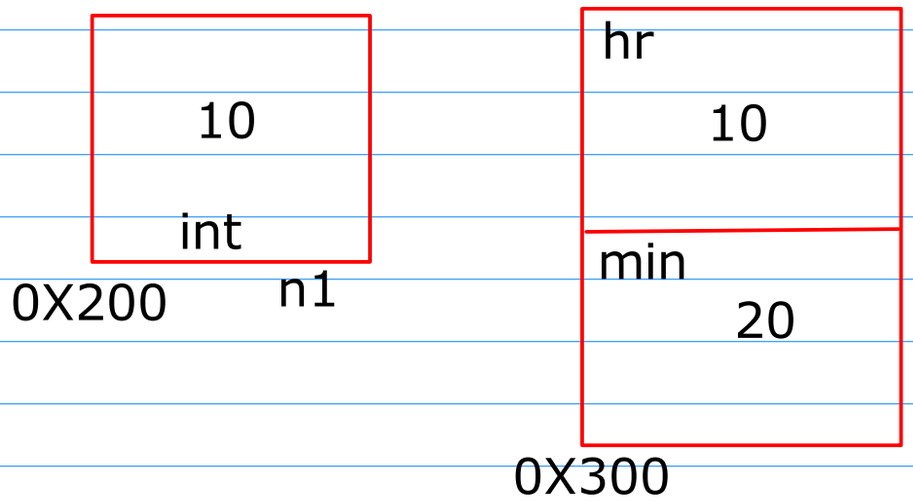
Dynamic Memory Allocation Text/Code Stack Heap Data

```
int n1;//stack
int *ptr = new int;

Time *ptr= new Time();
```




```
int main(){
        int n1 = 10;
        Time t1(10,20);
}
```




```
int main(){
        int *ptr = new int;
        *ptr = 10;


        Time*ptr2 = new Time(10,20);

        delete ptr;
        delete ptr2;
        ptr = NULL;
        ptr2 = NULL;
//To Avoid Dangling Pointers
}
```