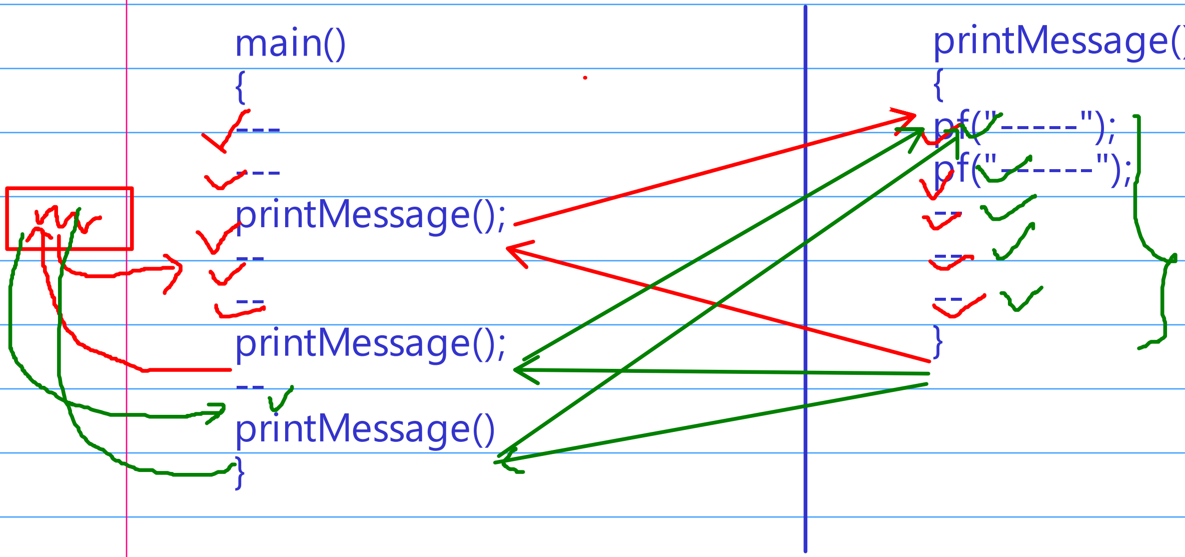
C++ => C + OOP

C++ => 100%

35 % => sec B

70%

FAR



```
main()
{
---
---
printMessage();
--
--
printMessage();
--
printMessage()
}
```

```
printMessage()
{
pf("-----");
pf("------");
--
--
--
}
```

#$

void printValue(int a) => printValue@int -zprintValuei

void printValue(int a,int b) => printValue@int,int -zprintValueii

void printValue(char a) => printValue@char

void printValue(int a,char c) => printValue@int,char

void printValue(char c,int a) => printValue@char,int

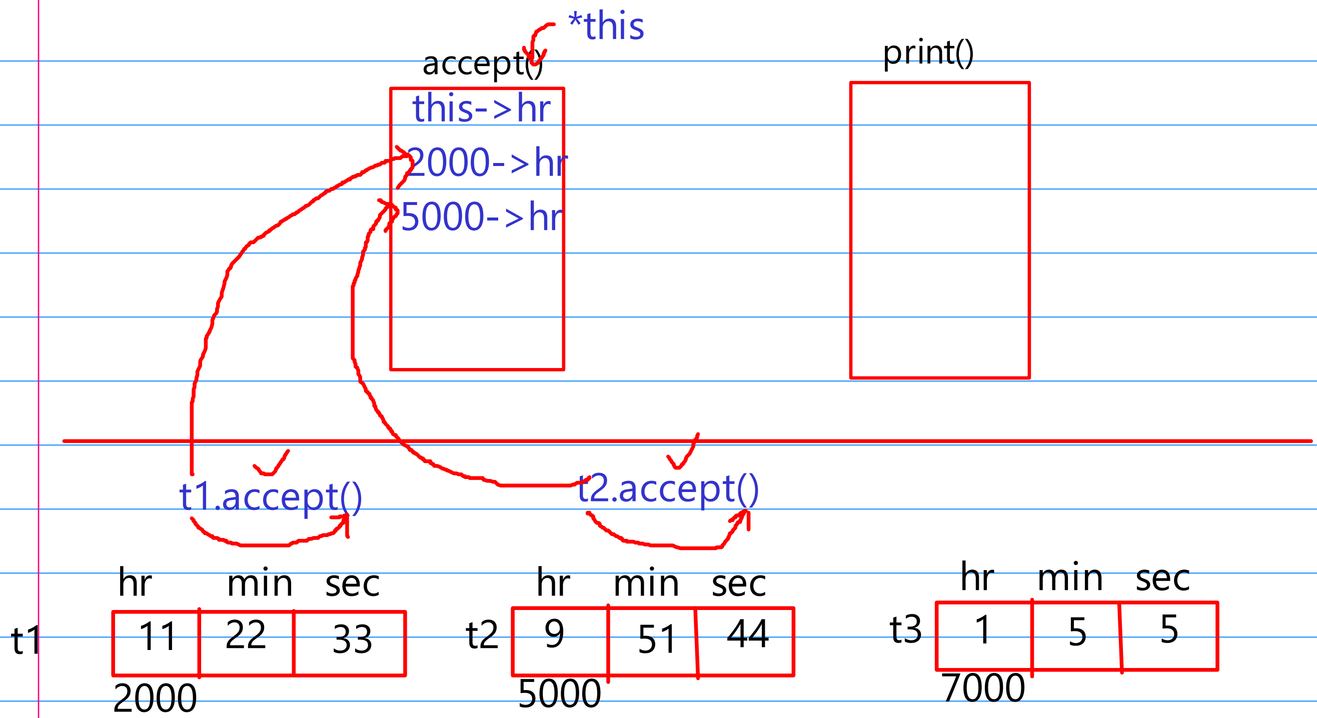
int -> 4 bytes => 32 bits

1 0 => 1 bit

bool => 1 byte => 8 bits

book
- price ✓
- auth ✓
- name ✓
- subj ✓
- pages ✓
- IDBI ✓
- year ✓
- versi ✓
- salary ✗
- roll_No ✗

time
- hr
- min
- sec

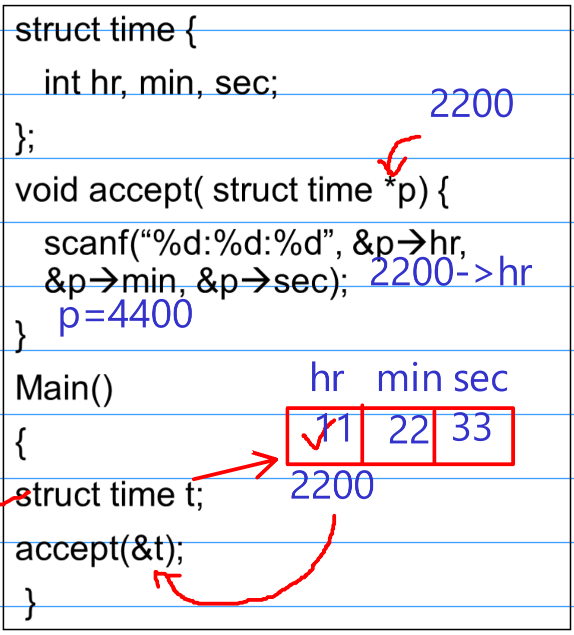


cpp => this
java => this
c# => this
python => self

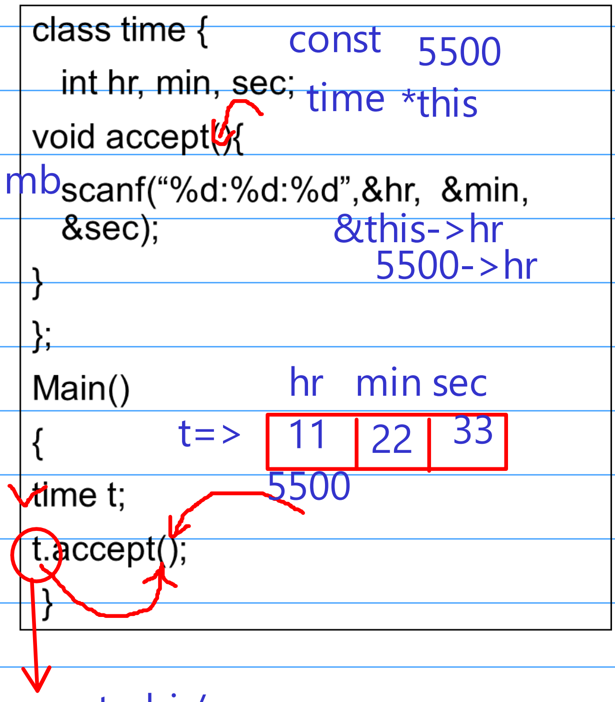
Structure in C

class in C++

gb



```
struct time {
   int hr, min, sec;
};
void accept( struct time *p) {
   scanf("%d:%d:%d", &p→hr,
   &p→min, &p→sec);
}
Main()
{
struct time t;
accept(&t);
 }
```



```
class time {
   int hr, min, sec;
void accept(){
   scanf("%d:%d:%d",&hr,  &min,
   &sec);
}
};
Main()
{
time t;
t.accept();
}
```

current obj /
calling obj

basic   app   req

int n1;

n1=10

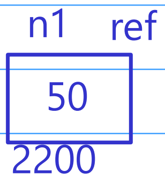
n1=15

int &ref = n1

ref = 50

cout<< ref => 50

cout<<n1 => 50



int a => int data type vari as a

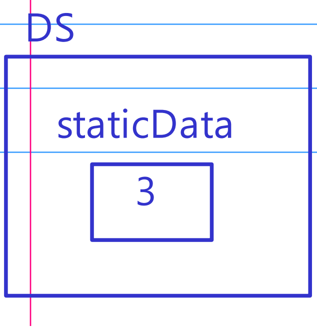
int *p =>int pointer type vari

int &r => int ref type vari

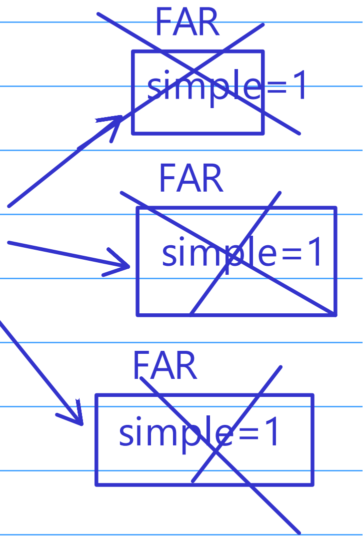
```
complex
{public:

sum(complex &c2)
{
 this-> c1
c2 => para
}

}
```

```
main()
{

  complex c1(5,7)
  complex c2(3,2)


 c1.real+c2.real //error

 c1.sum(c2)


}
```

```
void staticDemo()
{
    int simple=1;
    static int staticData;
        staticData++;
}
```



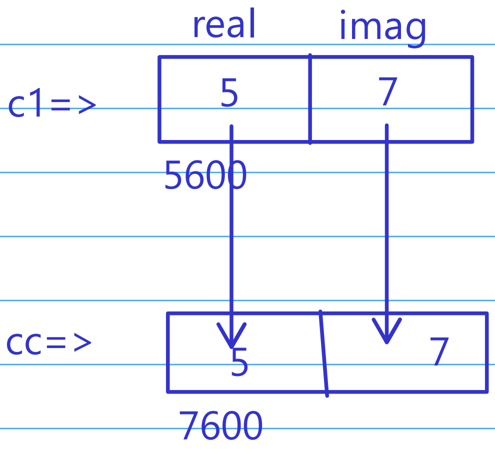
```
main()
{
        staticDemo();
        staticDemo();
        staticDemo();

}
```



account
accNo
rate_of_intr
3.6

a[10000000]

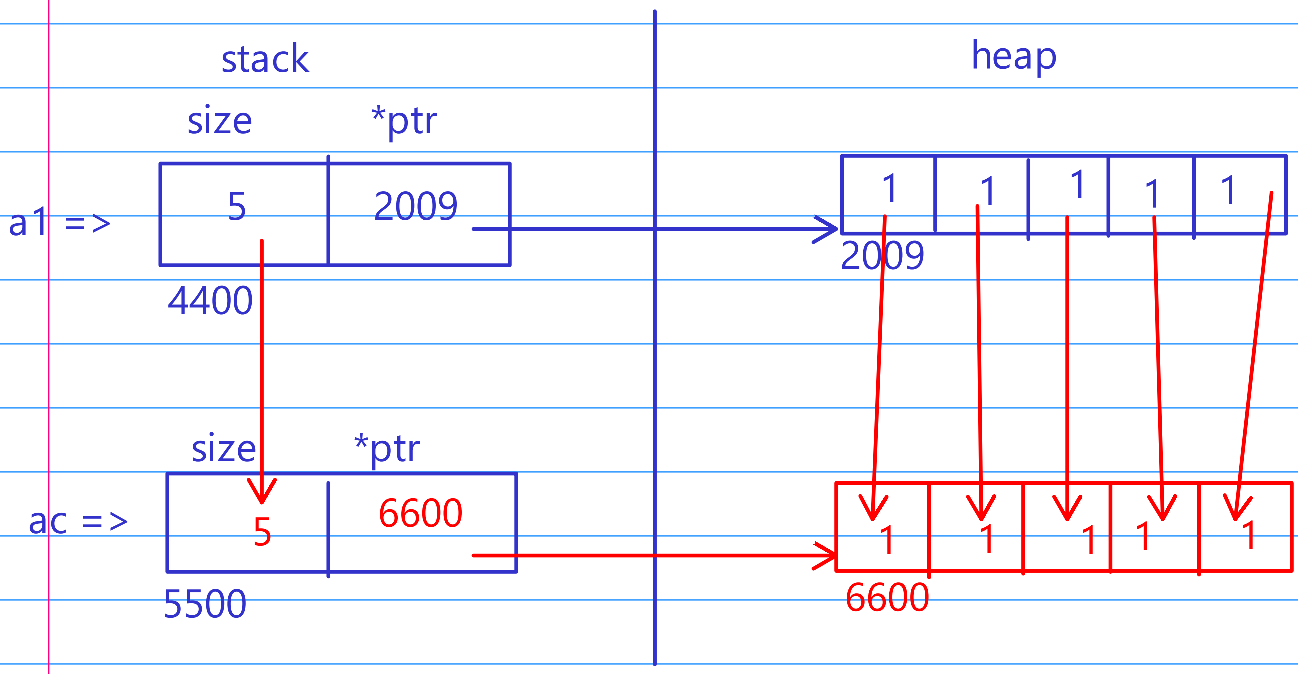
10000000X4
4

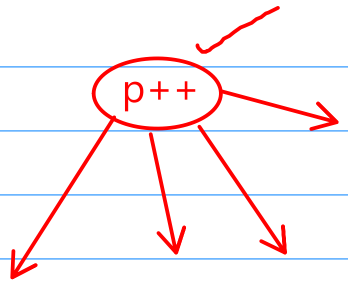


shallow copy

old case

new case





Abstraction

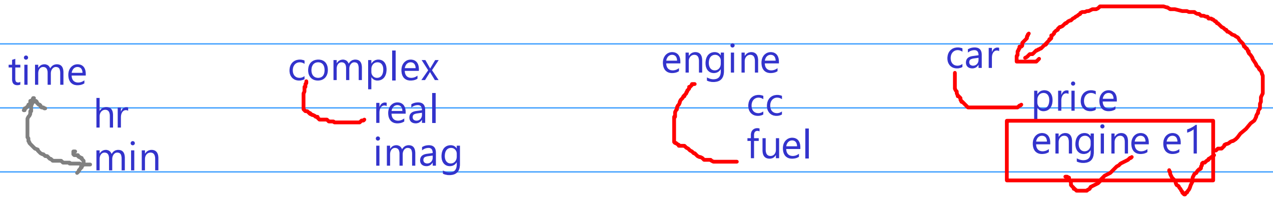
```
main()
{

printf("Enter %d",num);

        //calling fun
```

```
main()
{
account a1;
a1.deposite()
a1.withdrow()
```

Encapsulation

```
printf(-------)
{
--
--
--
--
--
--
--
}
```

```
class account
{
private:
accNo, PAN
bal,
name
--
--
fun1(){--}
fun2() {--}
pub:
deposite(){--}
withdrow(){--}
}
```

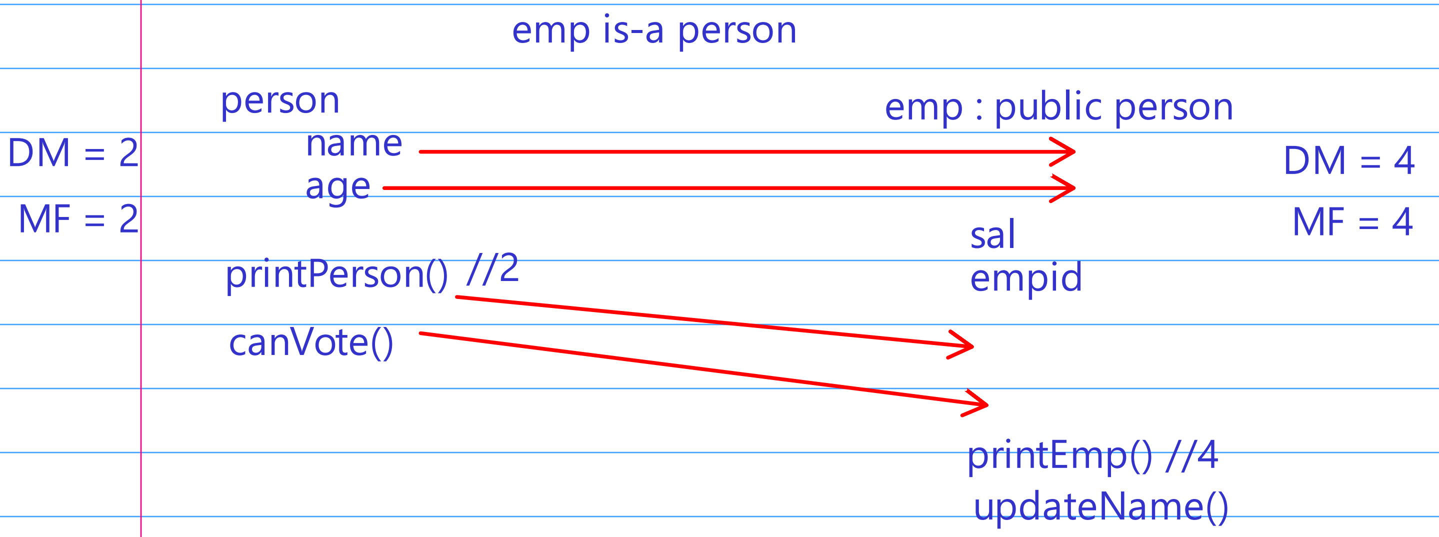


time
hr
min

complex
real
imag

engine
cc
fuel

car
price
engine e1



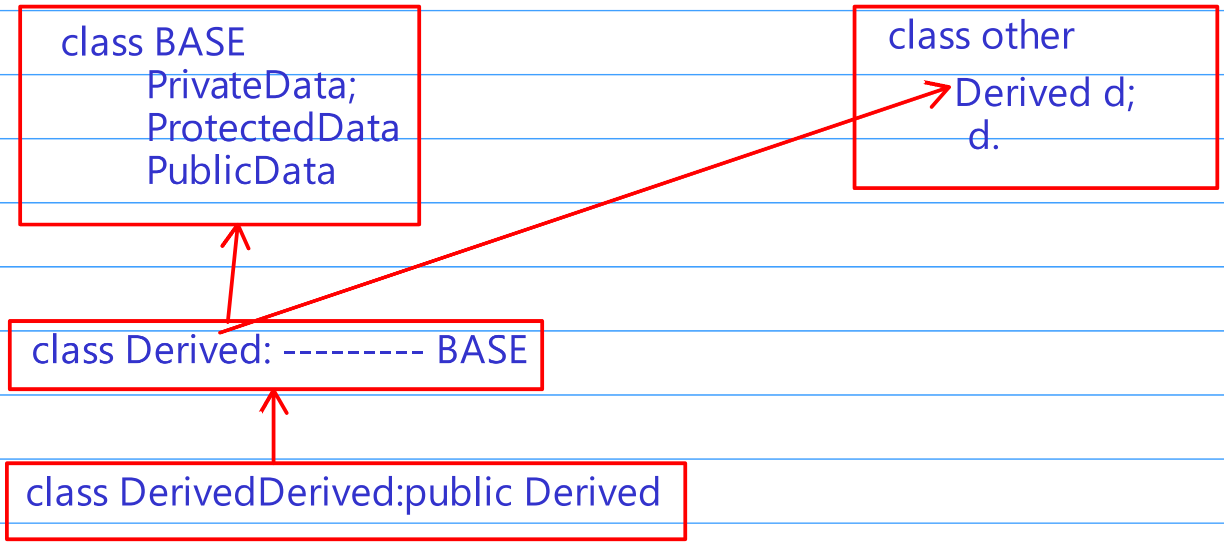
emp is-a person

person
name
age
printPerson() //2
canVote()

DM = 2
MF = 2

emp : public person
sal
empid
printEmp() //4
updateName()

DM = 4
MF = 4



class BASE
PrivateData;
ProtectedData
PublicData

class other
Derived d;
d.

class Derived: --------- BASE

class DerivedDerived:public Derived