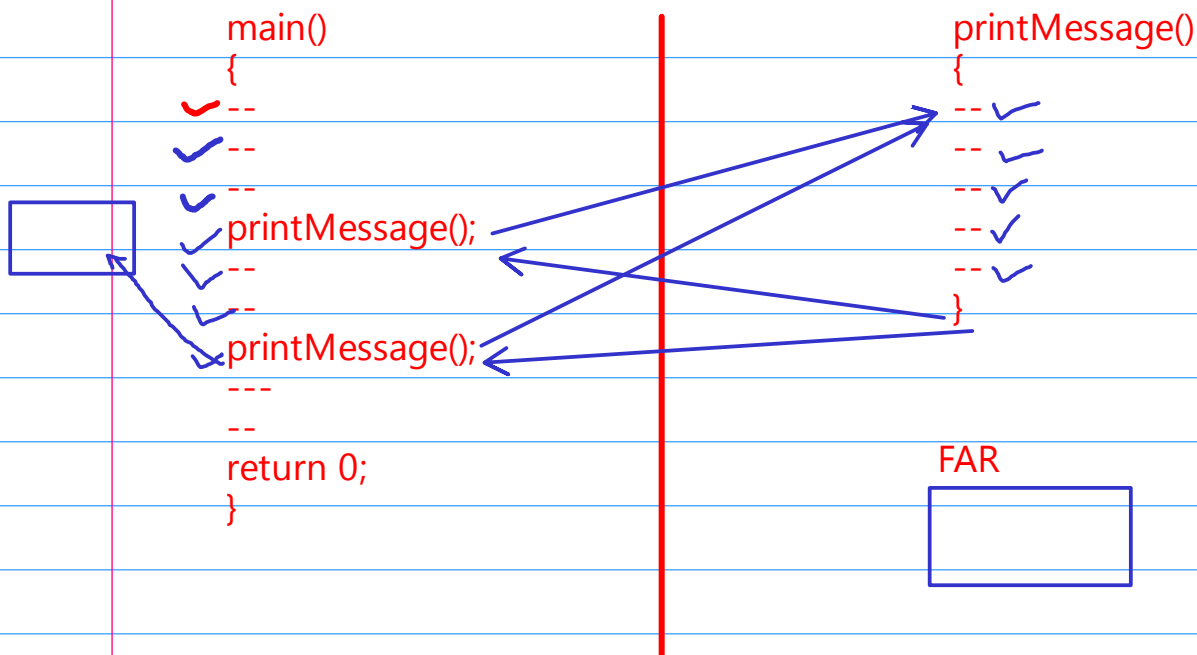


DAC cpp = 100%
preCAT CPP = 35%
DESD cpp = 70%

CPP= c+oop



#

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

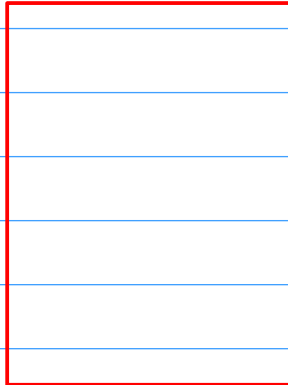
1 / 0 int => 4 => 32 1
bool => 1 => 8

char => 1 => 8 => 256
wchar_t => 16 => 2 => 65536

book ✓
pages ✓
author ✓
price ✓
pub ✓
sal
roll_no ✗

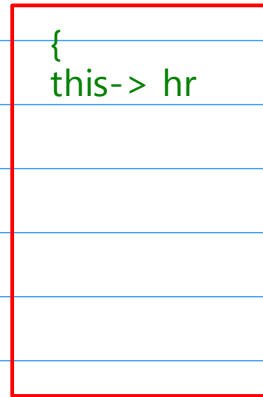
time
hr
min
sec

print()



accept() → this=4000

{
this-> hr



	hr	min	sec		hr	min	sec		hr	min	sec		
t1	11	22	33		t2	10	3	20		t3	8	5	33
	2000					3000					4000		

t1.accept()

t2.accept()

t3.accept()

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

Structure in C

```

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);
}
    
```

Diagram for C Structure:

hr	min	sec
11		

Annotations:
 - **7000** is written next to the struct definition and the variable `t`.
 - A red arrow points from `accept(&t);` to the struct definition.
 - A red arrow points from `&t` to the struct definition.
 - A red arrow points from `t` in the function call to the struct definition.

class in C++

```

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

Diagram for C++ Class:

hr	min	sec
12		

Annotations:
 - **const** is written next to the class definition.
 - **time * this** is written next to the `accept()` function.
 - **&this->hr** is written next to the `scanf` call.
 - A red arrow points from `t.accept();` to the `accept()` function.
 - A red arrow points from `t` in the function call to the class definition.
 - A red arrow points from `t` in the function call to the `accept()` function.

current obj
calling obj

basic

app

req

```

int n1
n1=12
n1=55
    
```

Diagram for basic example:

n1
66

Annotations:
 - **5500** is written below the variable `n1`.
 - A red arrow points from `n1` in the function call to the variable `n1`.

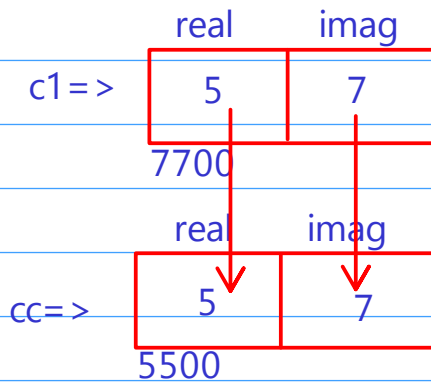
int& ref = n1

ref=66

n1=> 66

ref=> 66

n=> int type vari
 int* p=> int pointer type vari
 int& r=> int ref type vari



```

complex
{
  pub:
  sum(← complex& c2)
  {
    this->real += c2.real
  }
}

```

```

main()
{
  complex c1(7,6)
  complex c2(3,2)
  complex c3

  //c1.real+c2.real

  c1.sum(c2)
}

```

time

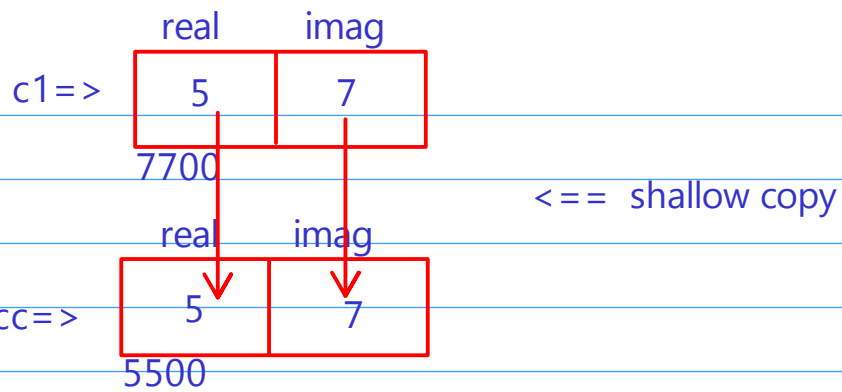
hr
min
sec

complex

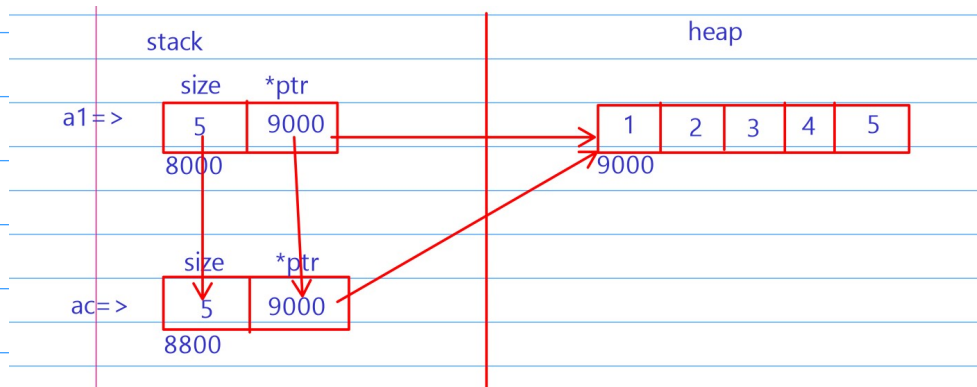
real
imag

car

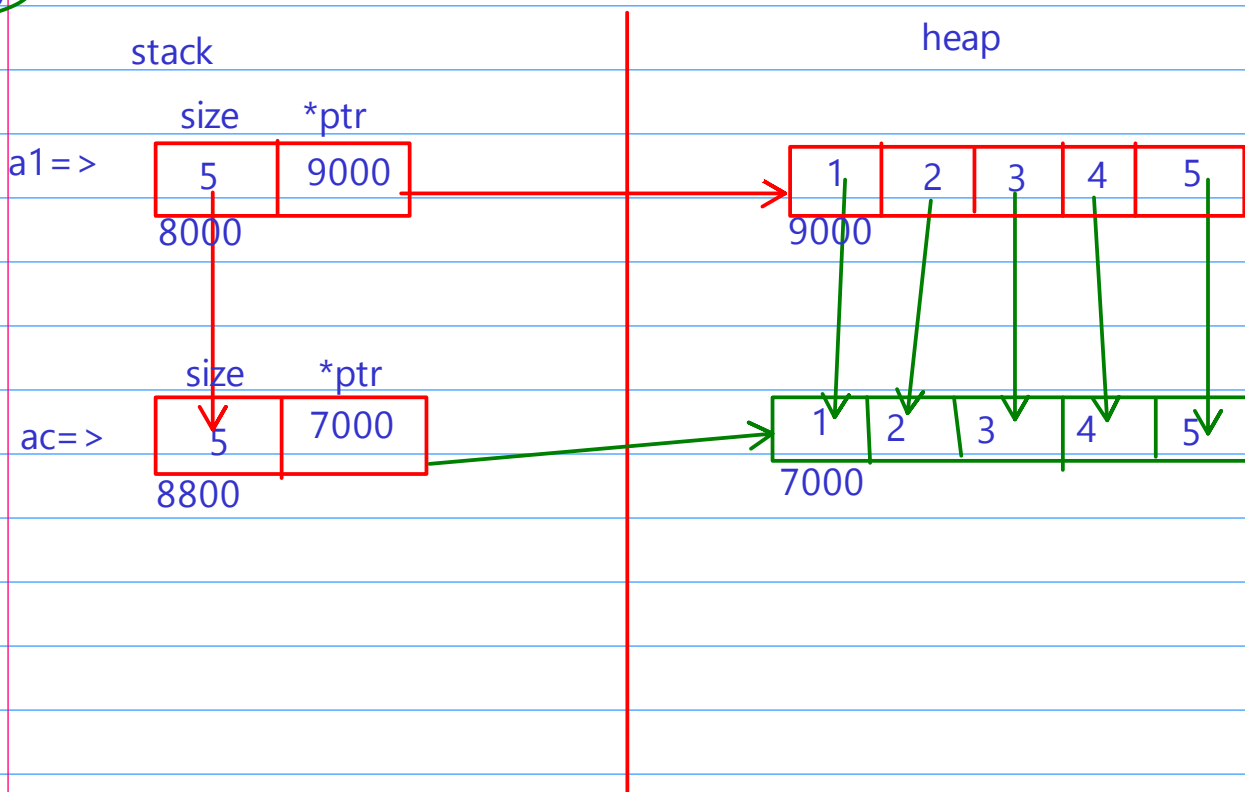
price
engine



shallow copy



Deep copy




Abstraction

```
main() {  
    printf("---",---);  
}
```

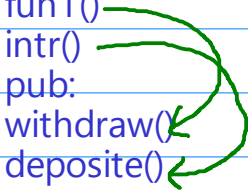

Encapsulation

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



```
main()  
{  
    account a1;  
    a1.  
  
}
```

```
class account  
{  
    private:  
        no.  
        name  
        email  
        bal  
        pan  
        fun1()  
        intr()  
    pub:  
        withdraw()  
        deposit()
```



DM=2
MF=1

emp is-a person

DM=4
MF=3

```
person  
{
```

```
  name
```

```
  age
```

```
  printPerson() //2
```

```
}
```

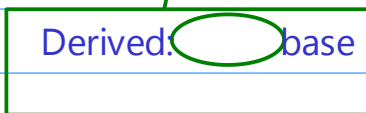
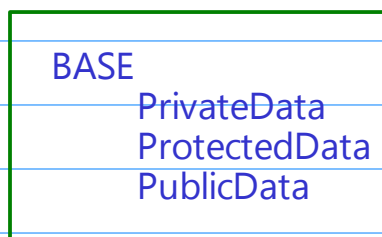
```
emp : public person  
{
```

```
  empid
```

```
  sal
```

```
  printEmp()//4  
  updateName()
```

```
}
```



class other
Derived d