OOPs → Classes & objects

⇒ e-commerce appln
customer
↳ age ✓
↳ name ✓
↳ gender ✓
↳ credits ✓
} Info.

⇒ maintaining an array for each attr

```
class Customer {
    public: → access specifier
        int age;
        char name[100];
        char gender;
        double credits;
};
```

body

Soln is switch to OOPs → Classes & objects

Blueprint of entity
↳ attributes ✓
action/behaviour

⇒ Customer class
⇒ age
name
credits
gender

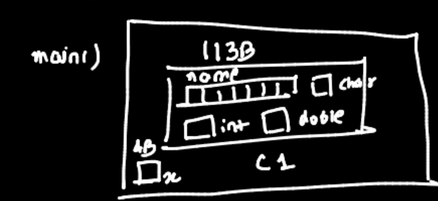
✓ shop()
✓ applyCoupon
✓ setAge
✓ giveRating

```
int main() {
    Customer C1;

    int x;

}
```

Stack



100B ✓
4B ✓
8B ✓
1B ✓
113B

```
int main( ) {
    customer c1;
    cout << c1.age << endl;   // garbage
    int x;
    cout << x;
}
```

'.' dot operator

```
int main() {
    customer c1;
    c1.age = 20;
    c1.gender = 'M';
    c1.credits = 100;
    strcpy(c1.name, "Ablishek");
    c1.name = "Ablishek"
}
```

dst

source

```
int main() {
    customer c1;
    }
```

→ memory is allocated

↓

who? compiler

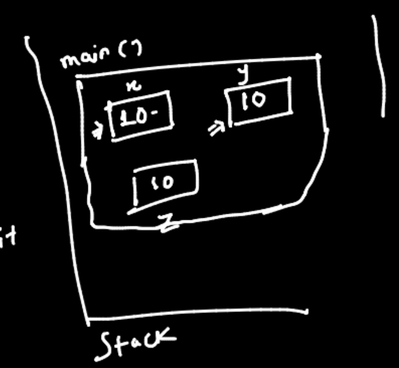
⇓

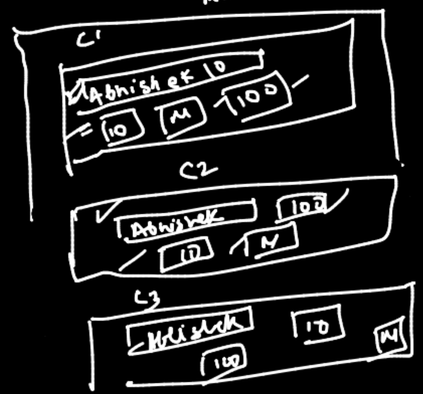
⇒ constructor

default constructor

public special member fn of the class that has same names as class name, no return type, no arguments

```
int main() {
    int x = 10;
    int y = x;   // init
    int z;   // decl.
    z = x;   // assgn.
}
```



```
int main() {
    Customer c1("Abhishek", 10, 'M', 100);
    Customer c2 = c1; <- copy an object
    Customer c3(c1);
}
```

default
copy
constructor