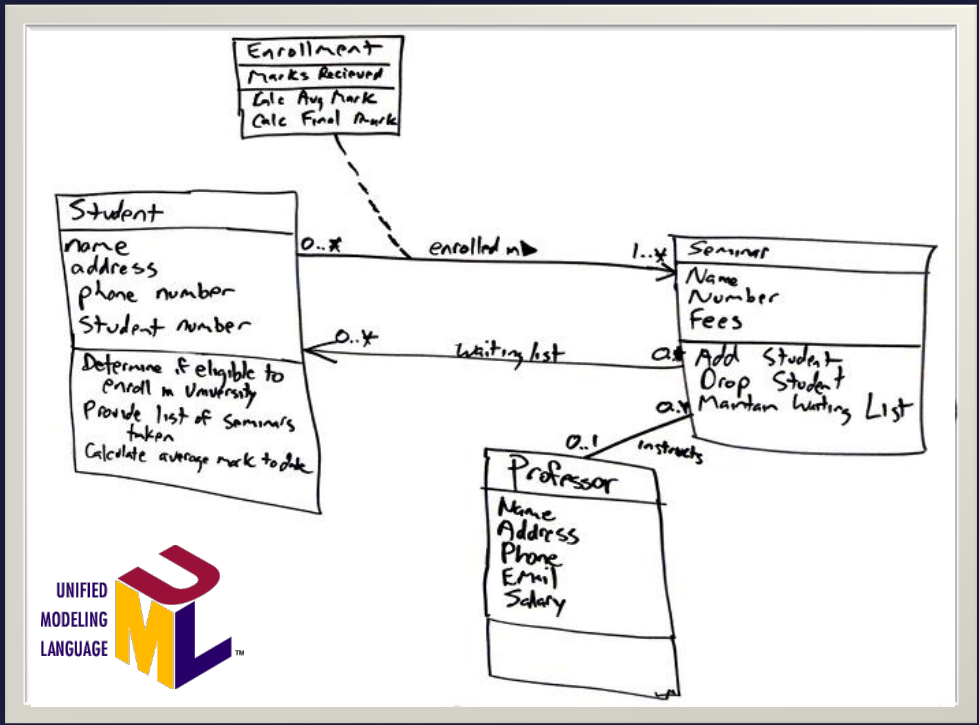
دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر

وراثت و چندریختی — ۱

مروری بر مفاهیم پایه شیءگرایی

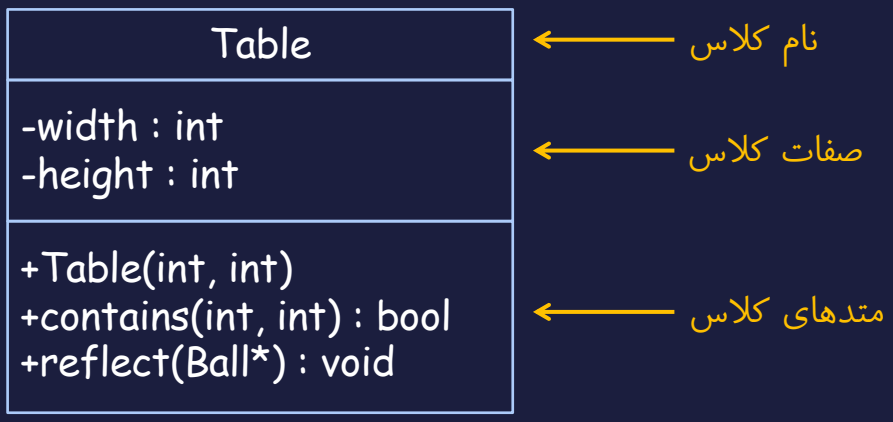
برنامه‌سازی پیشرفته — رامتین خسروی

بهار ۹۹

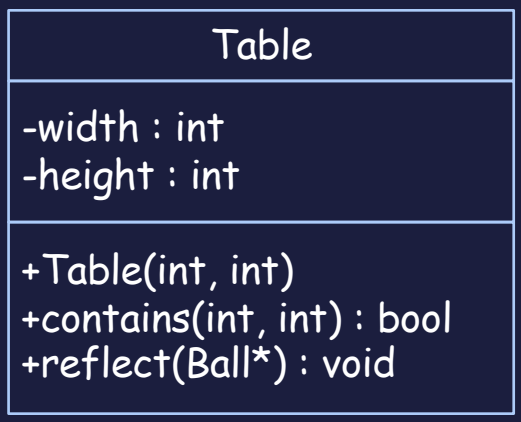


نمایش کلاس‌ها در UML

نمایش کلاس‌ها در یوام‌ال



کلاس‌ها در یوام‌ال و کد برنامه




```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
  void reflect(Ball*);
private:
  int width;
  int height;
};
```

نمایش‌های خلاصه‌تر

Table
-width : int -height : int
+Table(int, int) +contains(int, int) : bool +reflect(Ball*) : void

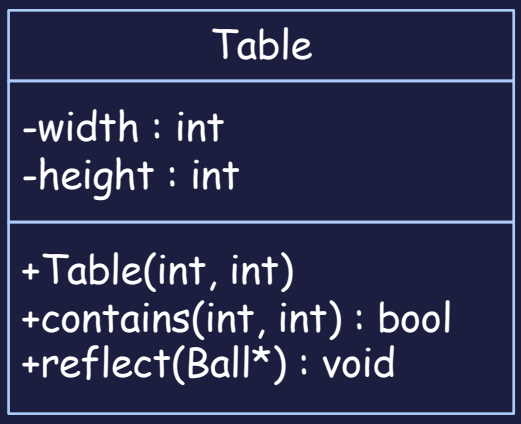
Table
width height
Table() contains() reflect()

نمایش‌های خلاصه‌تر

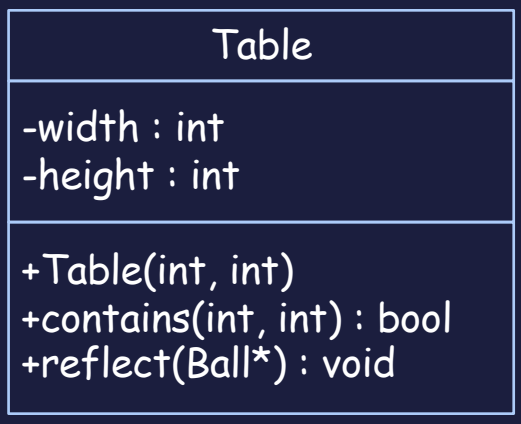
Table
-width : int -height : int
+Table(int, int) +contains(int, int) : bool +reflect(Ball*) : void

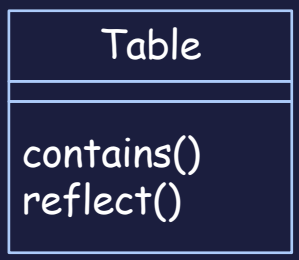
Table
width height
contains() reflect()

نمایش‌های خلاصه‌تر

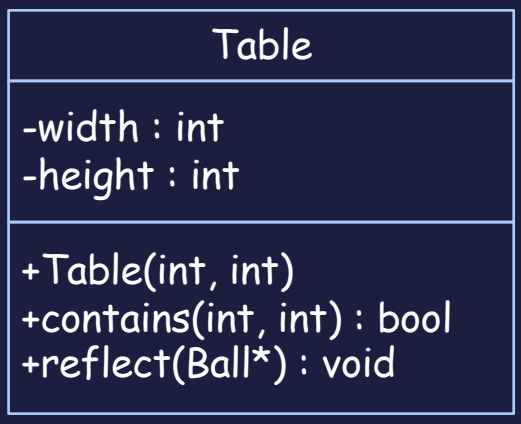


نمایش‌های خلاصه‌تر

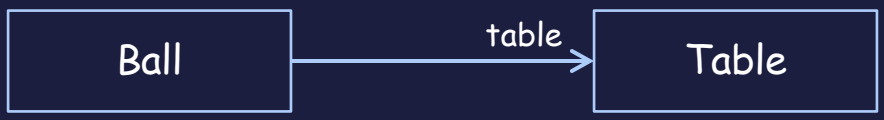




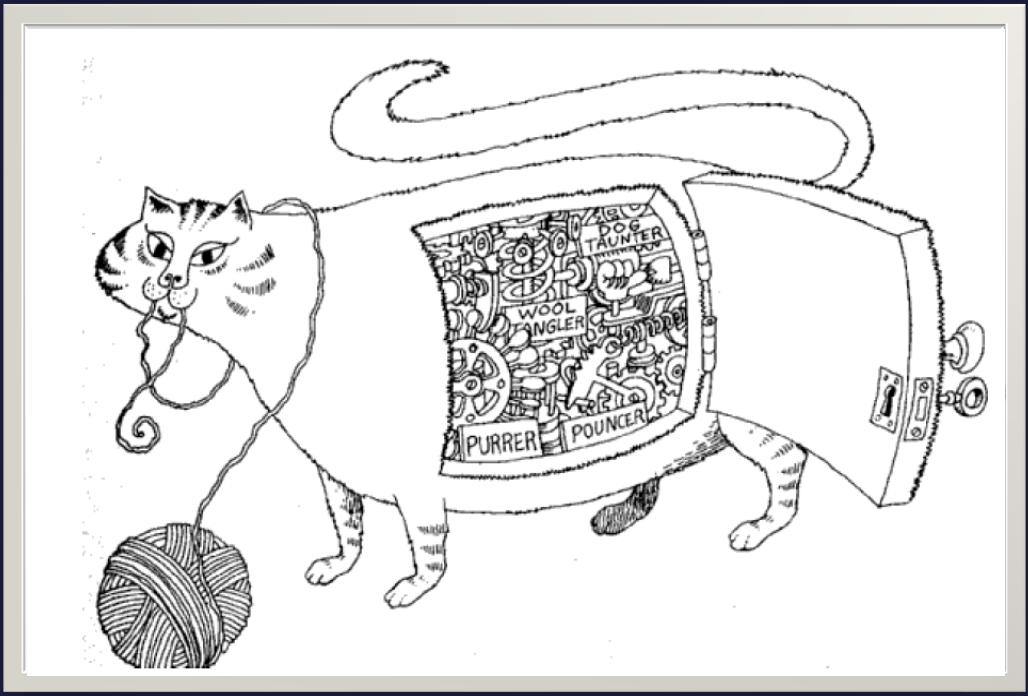
نمایش‌های خلاصه‌تر



نمایش روابط بین کلاس‌ها




```
class Ball {
  ...
private:
  Table* table;
};
```



encapsulation

کپسول‌سازی

Image taken from "Object-Oriented Analysis and Design", 3rd ed., by Grady Booch et al.

11

```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
  void reflect(Ball*);
private:
  int width;
  int height;
};
```

کنار هم قرار گرفتن داده‌ها و عملیات به عنوان موجودیتی واحد

12

```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
  void reflect(Ball*);
private:
  int width;
  int height;
};

Table::Table(int w, int h) {
  ...
}

// body of other methods
```

جداسازی واسط (interface) از پیاده‌سازی (implementation)


```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
  void reflect(Ball*);
private:
  int width;
  int height;
};

Table::Table(int w, int h) {
  ...
}

// body of other methods
```

واسط کلاس Table

واسط یک کلاس مانند قراردادی است که طبق آن می‌توان از آن کلاس استفاده کرد.

protocol = قرارداد

```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
  void reflect(Ball*);
private:
  int width;
  int height;
};

Table::Table(int w, int h) {
  ...
}

// body of other methods
```

پیاده‌سازی یک کلاس برای کلاس‌های دیگر غیرقابل دسترسی است.

پیاده‌سازی کلاس Table

مخفی‌سازی اطلاعات

یک کلاس باید تمام اطلاعاتی را که برای استفاده از آن لازم است به بیرون عرضه کند؛ و نه بیشتر!

```
class Table {
public:
  Table(int, int);
  int width;
  int height;
};
```

میزی مستطیل که ابعاد آن در دو متغیر به نام‌های width و height نگهداری می‌شوند.

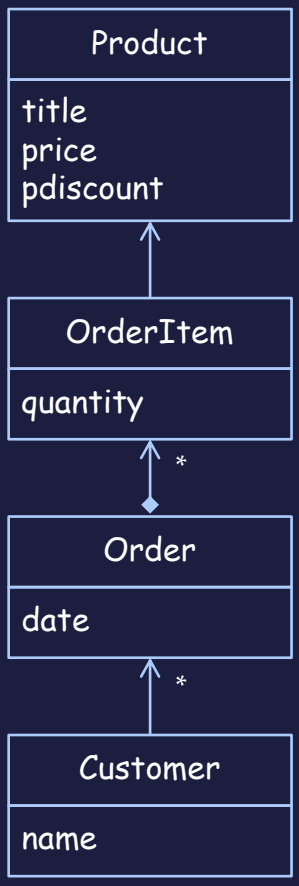
```
class Table {
public:
  Table(int, int);
  int get_width();
  int get_height();
private:
  int width;
  int height;
};
```

میزی مستطیل که می‌توان ابعاد آن را از آن پرسید.

```
class Table {
public:
  Table(int, int);
  bool contains(int, int);
private:
  int width;
  int height;
};
```

میزی که ... ؟

تمرین عملی



مسئله: محاسبه کل بدهی یک مشتری

- هر محصول می‌تواند درصدی تخفیف داشته باشد.
- خریدهای بیش از ۲ عدد از یک محصول نصف قیمت محسوب می‌شوند.

راه‌حل ارائه‌شده در اسلایدهای بعد کپسول‌سازی را رعایت نکرده است. کد را به نحو مناسبی تغییر دهید.

کد مثال در OrderMgmt.cpp

```cpp
class Product {
public:
    string get_title() { return title; }
    int get_price() { return price; }
    int get_percentage_discount() { return pdiscount; }
private:
    string title;
    int price;
    int pdiscount;
};
```

19

```cpp
class OrderItem {
public:
    Product* get_product() { return product; }
    int get_quantity() { return quantity; }
private:
    Product* product;
    int quantity;
};

class Order {
public:
    vector<OrderItem> get_items();
private:
    string date;
    vector<OrderItem> items;
};
```

20

```cpp
class Customer {
public:
    int total_bedehi();
private:
    string name;
    vector<Order*> orders;
};
```

21

```cpp
int Customer::total_bedehi() {
  int total = 0;
  for (Order* order : orders) {
    int order_total = 0;
    for (OrderItem order_item : order->get_items()) {
      Product *product = order_item.get_product();
      int unit_price = product->get_price();
      int pdiscount = product->get_percentage_discount();
      int quantity = order_item.get_quantity();
      order_total += unit_price*pdiscount/100 * quantity;
      if (quantity > 2 && unit_price > 50000)
        order_total -= (quantity - 2) * unit_price / 2;
    }
    total += order_total;
  }
  return total;
}
```

22