

APOLLO

B3 Object-oriented programming HL

Computer Science
First assessment 2027
2 questions | mark schemes

3. (a) (i) Award [3 max]

The `(Passenger)` constructor is called;

A new object is created / memory is assigned for a new object ...;

... identified by the variable `p1`;

The `passengerName`, `email` and `ticketPrice` variables / all of the variables are initialized;

(ii) Award [2 max]

provides greater security for the values stored in these variables/for the code;

restricts access to such variables from outside of the class /only allows access through public / getter, setter methods;

(helps to) prevents accidental/unintended changes to the variables (from outside of the class);

(b) Award [2 max];

composition;

The variable `ticketPrice` in the `Passenger` class depends upon the `Flight` class;

(c) Award [4 max]

Award [1] for calculating the % seats sold;

Award [1] for all 3 conditions correct;

Award [1] for if ... else structure correct;

Award [1] for return;

```
public double updatedTicketPrice()
{
    double price;
    double percentSold = (seatsBooked*100)/maxPassengers;
    if (percentSold >= 10 && percentSold <= 50)
        price = initialPrice*1.5;
    else
        if (percentSold > 50)
            price = initialPrice*3;
        else
            price = initialPrice;
    return price;
}
```

– 11 –

(d) **Award [4 max]**

Award [1] for calling `updatedTicketPrice()`;

Award [1] for creating new `Passenger` **object**;

Award [1] for updating `passengers` **array**;

Award [1] for incrementing `seatsBooked`;

```
public void addPassenger(String name, String email)
{
    double s = updatedTicketPrice();
    Passenger p = new Passenger(name, email, s);
    passengers[seatsBooked] = p;
    seatsBooked = seatsBooked + 1;
}
```

(e) **Award [5 max]**

the `name` field of each `Passenger` object is used in the sorting process;

the array is divided into two parts: a sorted subset and an unsorted subset;

the smallest element is found in the unsorted part of the array and compared to the first (leftmost) element in the unsorted set;

if smaller, the two objects are swapped (and the first element in the unsorted set joins the sorted set);

the two steps above are repeated until the whole array is sorted / there are no more elements in the unsorted set;

9. (a) (i) Award [3 max]

The `(Passenger)` constructor is called;

A new object is created / Memory is assigned for a new object ...;

... identified by the variable `p1`;

The `passenger_name`, `email` and `ticket_price` variables / all of the variables are initialized;

(ii) Award [2 max]

provides greater security for the values stored in these variables/for the code;

restricts access to such variables from outside of the class /only allows access through public / getter, setter methods;

(helps to) prevents accidental/unintended changes to the variables (from outside of the class);

(b) Award [2 max]

composition;

The variable `ticket_price` in the `Passenger` class depends upon the `Flight` class;

(c) Award [4 max]

Award [1] for calculating the % seats sold;

(allow `len(self.passengers)` for `self.seats_booked`)

Award [1] for all 3 conditions correct;

Award [1] for if ... else structure correct;

Award [1] for return;

```
def updated_ticket_price(self):
    percent_sold = self.seats_booked * 100 / self.max_passengers
    if percent_sold >= 10 and percent_sold <= 50.0:
        price = self.initial_price * 1.5
    elif percent_sold > 50.0:
        price = self.initial_price * 3
    else:
        price = self.initial_price
    return price
```

(d) **Award [4 max]**

Award [1] for calling `updated_ticket_price()`;

Award [1] for creating new `Passenger` **object**;

Award [1] for updating `passengers` **list**;

Award [1] for incrementing `seats_booked`;

```
def add_passenger(self, name, email):  
    s = self.updated_ticket_price()  
    p = Passenger(name, email, s)  
    self.passengers.append(p)  
    self.seats_booked = self.seats_booked + 1
```

(e) **Award [5 max]**

the `name` field of each `Passenger` object is used in the sorting process;

the list is divided into two parts: a sorted subset and an unsorted subset;

the smallest element is found in the unsorted part of the list and compared to the first (leftmost) element in the unsorted set;

if smaller, the 2 objects are swapped (and the first element in the unsorted set joins the sorted set);

the two steps above are repeated until the whole list is sorted / there are no more elements in the unsorted set;