Review Midterm Notes with IAttendee\_B

1) Attendee (String name, String, address)

{ this.name = name;

this.address = address;

}

SelectiveAttendee (String name, String, address)

{ super(name, address)

sessionList = new ArrayList <Session> ( ) ;

// should also add one session here since multiplicity is 1..\*

// could have the constructor look like :

// SelectiveAttendee (String name, String, address, Session s1)

// then would have sessionList.add(s1) ;

)

|

FullAttendee (String name, String, address)

{ super(name, address)

|

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2) void getTotalPayments( )

{ double total = 0;

for (IAttendee att : attendeeList)

{ total += att.getRegistrationFee( ) // Polymorphism Here!

}

////return total;

S.O.P (“Total payments = “ + total ;

} // Ask What if list is empty??

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3) In SelectiveAttendee :

double getRegistrationFee ( )

{ double total = 0;

for (Session s1 : sessionList)

{ total += s1.getSessionFee() // Polymorphism Here!

}

return total;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4) Double smallestFee ( ) // Check the Boundary conditions

Session smallestFee ( ) ; // In Red IF you want to return

// the whole Session Object,

{ double small;

Session temp ;

If ( sessionList = null) { return null; } // or Double.Max\_Value if

// returning a primitive value double

If ( sessionList.size( ) = 0) { return null; }

small = sessionlist.get(0). getSessionFee ( ) ; // init small OR

temp = sessionlist.get(0) ;

// small = Double.Max\_Value ; // init small

for (Session s1 : sessionList)

{ if (s1.getSessionFee ( ) < small)

temp = s1 ;

{ small = s1.getSessionFee ( ) ) }

} // end of for loop

return small ; // after the for loop

return temp ;

}

Collections.sort (IAttendeeList, myComparator) // Another way to do.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5) addSession method in SelectiveAttendee.

void addSession (Session s2)

{ sessionList.add(s2) }

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6) How can the FullAttendee class have access to the fullConferenceFee data member that is in the Conference class? getFullFee method returns this fullConferenceFee

One possibility : Pass the Conference object into the FullAttendee constructor.

FullAttendee can have a data member Conference c1;

new FullAttendee ( “Joseph”, “100 main”, **this)** // pass it into the constructor

OR, A setter - setConference method inside of FullAttendee could get the Conference object inside.

Then we could :

double getRegistrationFee ( )

{ return c1. getFullFee ( ) ;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GO TO BED EARLY!!!!