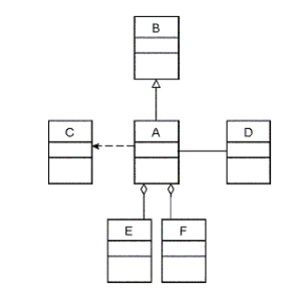
**UML diagrams. Practice.**

1. What is the relationship between class

A and class C?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A and class D?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A and class B?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A and class E?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

A and class F?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

B and class F?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Which of the following is not one of the defining features of the object-oriented model?

a. inheritance b. association c. encapsulation d. polymorphism

1. Which relationships mean "is a" ? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Which relationships mean "has a" ? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. What is the relationship between class a and class c \***Picture of black arrow pointing from class a to class c with a dashed line**\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Fill up the table:

|  |  |  |
| --- | --- | --- |
|  | Aggregation | Dependency |
| Life time |  |  |
| Child object |  |  |
| Relationship |  |  |
| Example |  |  |

You can use same words several times. Not all words will be used.

Child object doesn’t belong to a single parent

Have their own life

Child object belong to a single parent

Has-a

Is –a

Owner's life time

Uses

Car and Driver

Car and wheels

Person and Teacher

7. Which sentences are true?

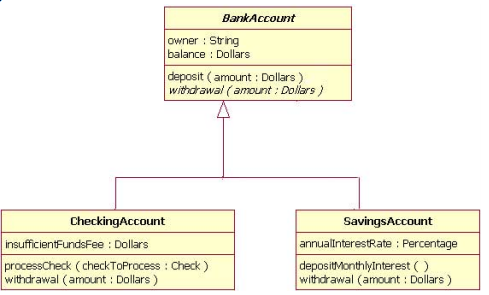
b) CheckingAccount and SavingAccount are BankAccount

c) CheckingAccount and SavingAccount are associated

d) BankAccount is associated to CheckingAccount

e) SavingAccount can processCheck

f) CheckingAccount has a balance



8. (a) What are relationships between these classes?

Car

-maxPassengers: int

-numPassengers: int

+Car(maxPass: int)

+getNumPassengers( ) : int

+addNumPassenger(num: int) : boolean

+removePassenger(num: int): void

Vehicle

+manufactuer: String

+model: String  
-year: short

-colour: String

-gasTankSize: float

-gas: float

-km: int

+Vehicle(manu: String, model: String,   
 year: short, colour: String,   
 tankS: float)

+paint(colour: String) : void  
+getColour(): String

+gasUp(litres:int): void

+drive(distance:int): Boolean

+getYear(): short

+getTankSize(): float

+getGasLeft(): float

+getKM(): int

CubeVan

-capacity: double

-maxCapacity: double

+CubeVan(maxCapacity: double)

+fillCubeVan(fill: double ) : boolean

+emptyCubeVan( ) : void

+currentCapacity( ) : double

Motorcycle

-engineSize: double

+MotorCycle(sizeEngine: double)

+getEngineSize ( ): double

(b) From the Motorcycle class, how would you call the PARENT class’ constructor? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(c) From the Car class, how would you call the PARENT class’ method getKM()? **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

9. What are relationships between these classes?

Animal

-name: String

-species: String

+age: int

-noise: String

+Animal(nam: String, spec: String,   
 a: int, noi: String)

+ printInfo(): String

+talk(): void

Zoo

-name: String

-location: String

+animals: Animal[]

+insects: Insect[]

+Zoo(nam: String, loc: String)

+printInfo(): void

+addAnimal(name: String, species: String, age: int, noise: String): void

+addInsect(n: String, nlegs: int, age: double, wings: boolean): void

Insect

-name: String

-numOfLegs: int

+age: int

-wings: boolean

-noise: String

+Insect(n: String, nlegs: int, a: int, w: boolean)

+ printInfo(): String

+makeNoise(): void

10. What are relationships between these classes?

Course

-code: String

-name: String

-level: String

Course(code: String, name: String,

level: String)

+getCode(): String

+getName(): String

+getLevel(): String

+toString():String

Student

-fname: String

-lname: String

-studentNumber: int

-course: EnrolledCourse[2]

-school: School

Student(sNum:int, first: String, last: String, s: School)

+getStudentNumber(): int

+addCourse(course:Course): boolean

+delCourse(course:Course): boolean

+getName(): String

+getSchool(): School

+getCourses():Course[]

+toString():String

1. 11. Convert each to UML diagram

School

-name: String

-address: String

-capacity: int

School(name: String, add: String, capacity:int)

+getName():String

+getAddress():String

+toString():String

EnrolledCourse

+course: Course

+absences: byte

+lates: byte

+mark: float

+status: String

EnrolledCourse(c: Course, a:byte, l: byte,

mark: float, s: String)

+getStatus(): String

+getLates(): byte

+getAbsences():byte

+addLates(num: byte) : void

+addAbsences(num: byte) : void

+setMark(mark: float): boolean

+setStatus(status: String): boolean

|  |
| --- |
| public class Vehicle {  public Wheel wheels[4];  public Person owner;  public String manufactuer;  public String model;  public short year;  private int km;  public Vehicle(String man, String mod, short y){…}  public void setOwner(Person p) {…}  public void addWheel(String b, String m, int s, String sea) {…}  private boolean drive(int distance){…}  public int getKM(){…}  } |
| public class Wheel {  public String brand;  public String model;  public int size;  private String season;  private String tread\_condition;  public Wheel(String b, String m, int s, String sea){…}  public String getSeason(){…}  private String getTreadCondition(){…}  } |
| public class Car extends Vehicle {  public int maxPassengers;  private int numPassengers;  public Car (String man, String mod, short y, int maxP){…}  public int getNumPassenger(){…}  public boolean addNumPassenger(int num){…}  public void removePassenger(int num){…}  } |
| public class Person {  public String name;  public long SIN;  public Date birthdate;    public Person(String nam, long sin, Date b){…}  } |