CS 1331 - Polymorphism WS

NOTE: THIS IS NOT A PRACTICE EXAM: It is not meant to in any way reflect the contents or format of Exam 3. This is a practice worksheet and is not meant to be the sole preparation for the exam. Questions on this worksheet are meant to give students a better understanding of course concepts for homeworks as well as future exams.

1.

a) Animal a = new Dog();
a.myName();
Compiles
“I am a Dog named Dog”

b) Object d = new Dog();
d.myName();
Does not compile

c) Object o2 = new Dog();
((Animal) o2).myName();
Compiles
“I am a Dog named Dog”

d) Dog d = new Animal();
d.myName();
Does not compile

e) Animal d = new Dog();
d.bark();
Does not compile

f) Object o2 = new Dog();
((Cat) o2).myName();
Compiles. Error occurs.

2.

Command	***Compiles?***	***Runs?***
Student student1 = new Student();	yes	yes
Student student2 = new Person();	no	no
Person person1 = new Student();	yes	yes
person1.where();	yes	yes
person1.studentMethod(student1);	no	no
((Student)person1).studentMethod(new CollegeStudent());	yes	yes
((Student)person1).studentMethod(student1);	yes	yes
(Student)person1.studentMethod(student1);	no	no
CollegeStudent collegeStudent0 = new CollegeStudent();	yes	yes
HighSchoolStudent castHS = (HighSchoolStudent) collegeStudent0;	no	no
CollegeStudent castCS = (CollegeStudent) student1;	yes	no

a) When casting an Object, when are we guaranteed that the casting will compile? When you cast to a superclass (or actual class) of the dynamic type.

b) When will a casting not only compile but also run without crashing the program? Again, casting to a superclass (or actual class) or the dynamic type.

c) Look at the equals method defined in the HighSchoolStudent class. Three mistakes were made in the implementation. Some of them will be caught at compile time, some of them are logical bugs. List the line where mistakes are found, and provide a short description of what's incorrect.

Line 2: When overriding the Object equals method, make sure to have the correct method signature. It should be **`public boolean equals(Object o)`**

Line 3: When checking for 2 objects being exactly the same, use `==`. Otherwise, if the passed in Object were a HighSchool object, you'll get infinite recursion from dynamic binding

Line 5: Static type should be HighSchoolStudent, not Student

3.

a) Animal a = new Bird();
   a.makeNoise();
   Compiles
   “Chirp chirp!”

b) Animal a = new Bird();
   a.fly();
   Does not compile

c) Animal a = new Bird();
   a.eat();
   Compiles
   “Nom”

d) Bird b = new Bird();
   Animal a = (Animal) b;
   Compiles

e) Animal a = new Animal( );
   Fish f = (Fish) a;
   Compiles. Error occurs

f) Animal a = new Fish();
   Fish f = (Fish) a;
   Compiles

g) Bird b = new Bird();
   Fish f = (Fish) b;
   Does not compile

h) Fish f = new Fish();
   Animal a = (Animal) f;
   Compiles

i) Animal a = new Bird();
   Bird b = (Bird) a;
   Compiles

j) Animal a = new Animal();
   Bird b = (Bird) a;
   Compiles. Error occurs

k) Animal a = new Fish();
a = (Bird) a;
Compiles. Error occurs

l) Bird b = new Bird();
((Animal) b).makeNoise();
Compiles
“Chirp chirp!”

m) Animal a = new Bird();
((Bird) a).fly();
Compiles
“I’m flying!”

n) Animal a = new Bird();
(Bird) a.fly();
Does not compile

o) Animal a = new Bird();
((Bird) a).eat();
Compiles
“Nom”

p) Animal a = new Bird();
((Bird) a).makeNoise();
Compiles
“Chirp chirp!”