Given the following coding segment:

```
public class Pet
  {
    public Pet()
    {
      //implementation not shown
    }
  }


public class Dog extends Pet
  {
    public Dog()
    {
      //implementation not shown
    }
  }


public class Cat extends Pet
  {
    public Cat()
    {
      //implementation not shown
    }
  }
```

Which of the following statements will compile and run without error? Select all that apply!

A.	`Dog dalmation1 = new Dog();`
	`Pet dalmation2 = (Pet)dalmation1;`

B.	`Pet dalmation3 = new Pet();`
	`Dog dalmation4 = (Dog)dalmation3;`

C.	`Cat si = new Cat();`
	`Cat am = si;`

D.	`Pet sgt = new Cat();`
	`Dog tibbs = (Dog)sgt;`

Given the following implementations:

```
public class Pet
  {
    public void meth1()
    {System.out.print("1");}
  }


public class Bunny extends Pet
  {
    public void meth2()
    {System.out.print("2");}
  }
```

Which of the following correctly identifies the output of each coding segment? Select all that apply.

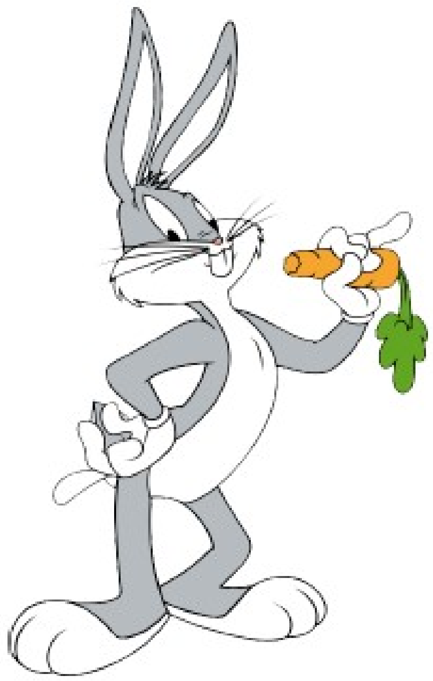
A.	`Bunny bugs1 = new Bunny();`
	`bugs1.meth1();`
		Will result in an error.

B.	`Pet bugs2 = new Bunny();`
	`bugs2.meth2();`
		Prints “2”

C.	`Pet bugs4 = new Pet();`
	`bugs4.meth1();`
		Prints “1”

D.	`Pet bugs = new Bunny();`
	`((Bunny)bugs).meth2();`
		Prints “2”

E.	`Bunny bugs5 = new Pet();`
	`bugs5.meth1();`
		Prints “1”



What will the following coding segment print to the screen?:

```
public class Pet
  {
    public void methOne()
    {
      System.out.print("A");
      methTwo();
    }

    public void methTwo()
    {
      System.out.print("B");
    }
  }


public class Dog extends Pet
  {
    public void methOne()
    {
      super.methOne();
      System.out.print("C");
    }

    public void methTwo()
    {
      super.methTwo();
      System.out.print("D");
    }
  }


public class Main {
  public static void main(String[] args) {
    Pet clifford = new Pet();
    clifford.methOne();
  }
}
```

Given the following coding segment:

```
public class Pet
  {
    public int x;
    public int y;

    public Pet()
    {}

    public Pet(int x, int y)
    {
      this.x = x;
      this.y = y;
    }
    //other methods
  }


public class Fish extends Pet
  {
    public int z;

    //other code

  }
```

Which of the following constructors would be valid for class Fish? Select all that apply!

A.	`public Fish(int z)`
	`{`
	`  this.z = z;`
	`  super(0,0);`
	`}`

B.	`public Fish(int x, int y, int z)`
	`{`
	`  super(x, y, z);`
	`}`

C.	`public Fish()`
	`{}`

D.	`public Fish(int x, int y, String z)`
	`{`
	`  super(x, y);`
	`  this.z = z.length();`
	`}`

E.	`public Fish(int x, int y)`
	`{`
	`  this.x = x;`
	`  this.y = y;`
	`  this.z = 0;`
	`}`

What will the following coding segment print to the screen?:

```
public class Pet1
  {
    public Pet1()
    {
      System.out.print("A");
    }
  }


public class Pet2 extends Pet1
  {
    public Pet2()
    {
      System.out.print("B");
    }
  }


public class Pet3 extends Pet2
  {
    public Pet3()
    {
      System.out.print("C");
    }
  }


public class Pet4 extends Pet1
  {
    public Pet4()
    {
      System.out.print("D");
    }
  }


public class Main {
  public static void main(String[] args) {
    Pet3 garfield = new Pet3();
  }
}
```

Given the following coding segment:

```
public class Pet
  {
    private int x;
    private int y;

    public Pet(int x, int y)
    {
      this.x = x;
      this.y = y;
    }
    //other methods
  }


public class Pony extends Pet
  {
    public int z;

    //other code

  }
```

Which of the following constructors would be valid for class Pony? Select all that apply!

A.	`public Pony(int z)`
	`{`
	`  this.z = z;`
	`  super(0,0);`
	`}`

B.	`public Pony(int x, int y, int a)`
	`{`
	`  super(x, y);`
	`  z = a;`
	`}`

C.	`public Pony()`
	`{}`

D.	`public Pony(int x, int y, String z)`
	`{`
	`  super(x, y);`
	`  this.z = z.length();`
	`}`

E.	`public Pony(int x, int y)`
	`{`
	`  this.x = x;`
	`  this.y = y;`
	`  this.z = 0;`
	`}`

What will the following coding segment print to the screen?:

```
public class Pet1
  {
    public Pet1()
    {
      System.out.print("A");
    }
  }


public class Pet2 extends Pet1
  {
    public Pet2()
    {
      System.out.print("B");
    }
  }


public class Pet3 extends Pet2
  {
    public Pet3()
    {
      System.out.print("C");
    }
  }


public class Pet4 extends Pet1
  {
    public Pet4()
    {
      System.out.print("D");
    }
  }


public class Main {
  public static void main(String[] args) {
    Pet4 scooby = new Pet4();
  }
}
```

Given the following implementations:

```
public class Human{. . .}
public class Artist extends Human {. . .}
public class Musician extends Artist {. . .}
```

Which of the following are legal statements? Select all that apply!

A.	`Human dojaCat = new Musician();`
B.	`Musician snoopDogg = new Artist();`
C.	`Human catStevens = new Artist();`
D.	`Artist catPower = new Musician();`

What will the following coding segment print to the screen?:

```
public class Pet
  {
    public void methOne()
    {
      System.out.print("A");
      methTwo();
    }

    public void methTwo()
    {
      System.out.print("B");
    }
  }


public class Dog extends Pet
  {
    public void methOne()
    {
      super.methOne();
      System.out.print("C");
    }

    public void methTwo()
    {
      super.methTwo();
      System.out.print("D");
    }
  }


public class Main {
  public static void main(String[] args)
    Pet snoopy = new Dog();
    snoopy.methOne();
  }
}
```

What will the following coding segment print to the screen?:

```
public class Pet
  {
    public void m1()
    {
      System.out.print("C");
    }

    public void m2()
    {
      System.out.print("E");
    }

    public String toString()
    {
      return "B";
    }
  }


public class Turtle extends Pet
  {
    public void m1()
    {
      System.out.print("D");
    }
  }


public class Main {
  public static void main(String[] args) {
    Turtle raphael = new Turtle();
    System.out.print(raphael);
    raphael.m1();
    raphael.m2();
  }
}
```

What will the following coding segment print to the screen?:

```
public class Pet
  {
    public void m1()
    {
      System.out.print("E");
    }

    public void m2()
    {
      System.out.print("D");
    }

    public String toString()
    {
      return "A";
    }
  }


public class Turtle extends Pet
  {
    public void m1()
    {
      System.out.print("C");
    }

    public void m2()
    {
      super.m1();
    }

    public String toString()
    {
      return super.toString() + "B";
    }
  }


public class Main {
  public static void main(String[] args) {
    Pet squirt = new Turtle();
    System.out.print(squirt);
    squirt.m1();
    squirt.m2();
  }
}
```