Dog

string name
date dob
string breed
date last_vacc_date
int weight
float ob_sch_gpa
int num_children_savaged

Cat

string name
date dob
string breed
float fluffiness
date last_vacc_date
float mousing_skill

Goldfish

string name
date dob
string shade

```
class pet
{ protected:
     string name;
     date dob;
  public:
    pet(string n, date d);
    void set_name(string x);
    string get_name();
    void set_dob(date x);
    date get_dob();
    virtual void print(); };


class goldfish: public pet
{ protected:
    string shade;
  public:
    goldfish(string n, string s);
    void set_shade(string x);
    string get_shade();
    virtual void print(); };


class mammal: public pet
{ protected:
    string breed;
    date last_vacc_date;
  public:
    mammal(string b);
    void set_breed(string x);
    string get_breed();
    void set_vac_date(date d);
    date get_vac_date();
    virtual void print(); };
```

```
class cat: public mammal
{ protected:
    float fluff, mousing;
    virtual void print();
  public:
    cat(string n, int d, string b);
    etc };


class dog: public mammal
{ protected:
    int weight;
    float ob_sch_gpa;
    int chil_sav;
  public:
    void print();
    etc };

struct date
{ int y, m, d;
  static date make_one_up;
  date(int y1, int m1, int d1); };

date::date(int y1, int m1, int d1)
{ y=y1; m=m1; d=d1; }

date date::make_one_up(999, 27, 42);


pet::pet(string n, date d)
{ name=n;
  dob=d; }

mammal::mammal(string n, date d, string b): pet(n, d)
{ breed=b; }
```

```
cat::cat(string n, date d, string b): mammal(n, d, b)
{ fluff = 0.0;
  mousing = 0.0; }

dog::dog(string n, date d, string b, int w): mammal(n, d, b)
{ weight  w;
  chil_sav = 0; }

goldfish::goldfish(string n, string s): pet(n, date::make_one_up)
{ shade = “gold”; }



void pet::print()
{ cout << “pet(“ << name << “,”;
  dob.print();
  cout << “)”; }


void mammal::print()
{ cout << “mammal(“;
  pet::print();
  cout << “,” << breed << “)”; }


void cat::print()
{ cout << “cat(“;
  mammal::print();
  cout << “,” << fluffiness << “)”; }


void dog::print()
{ cout << “dog(“;
  mammal::print();
  cout << “,” << weight << “,” << chil_sav << “)”; }
```

```
pet * everyone[10000];

cat  * c1 = new cat("Frank", date(2006, 11, 13), "Persian");
c1->print();
```

cat(mammal(pet(Frank, 13th Nov 2006), Persian), 0.0)

```
dog * d1 = new dog("Ratzo", date(2004, 1, 22), "Fleahound", 35);
d1->print();
```

dog(mammal(pet(Ratzo, 22nd Jan 2004), Fleahound), 0, 35)

```
goldfish * g1 = new goldfish("Yellowie", "Yellow");
g1->print();
```

goldfish(pet(Yellowie, unknown), Yellow)

```
everyone[3] = c1;
everyone[4] = d1;
everyone[5] = g1;

for (int i=3; i<=5; i+=1)
  everyone[i]->print();
```

WITHOUT the word virtual in the print declarations, we would see

pet(Frank, 13th Nov 2006)
pet(Ratzo, 22nd Jan 2004)
pet(Yellowie, unknown)

WITH the word virtual as shown, we would see

cat(mammal(pet(Frank, 13th Nov 2006), Persian), 0.0)
dog(mammal(pet(Ratzo, 22nd Jan 2004), Fleahound), 0, 35)
goldfish(pet(Yellowie, unknown), Yellow)

pet(Frank, 13th Nov 2006)

```
cout << c1->get_name();
cout << everyone[3]->get_name();
cout << c1->get_fluffiness();
cout << everyone[3]->get_fluffiness();   NONNONONONONONO
```

```
cat * c2;

pet *p2 = new pet("name", date(2007,1,1));

c2=p2;    NONONO

c2->set_fluffiness(0.5);
```