#include <iostream>

class CAT{
public:
    CAT();           //default constructor
    CAT (const CAT&);  //copy constructor
    ~CAT();          //destructor
    int GetAge() const {return *itsAge;}
    int GetWeight() const {return *itsWeight;}
    void SetAge(int age) {*itsAge = age;}

private:
    int *itsAge;
    int *itsWeight;
};

CAT::CAT(){
    itsAge = new int;
    itsWeight = new int;
    *itsAge = 5;
    *itsWeight = 9;
}

CAT::CAT(const CAT& rhs){   //deep copy
    itsAge = new int;
    itsWeight = new int;
    *itsAge = rhs.GetAge();
    *itsWeight = rhs.GetWeight();
}

CAT::~CAT(){
    delete itsAge;
    itsAge = NULL;
    delete itsWeight;
    itsWeight = NULL;
    std::cout << "destructor called\n";
}

int main(int argc, const char * argv[])
{
    CAT frisky;
    frisky.SetAge(45);
    CAT boots(frisky);  // created boots from frisky
    frisky.SetAge(3);
    std::cout << "boots is " << boots.GetAge() << std::endl;
    return 0;
    //when scope gone destructors are called
}