typedef struct {
    unsigned side;
} square;

void square_square(square *this, unsigned side) {
    this->side = side;
}

unsigned
square_area(square *this) {
    return this->side * this->side;
}

unsigned things_and_stuff() {
    square s;
    square_square(&s, 32);
    unsigned area = square_area(&s);
    // destructor call
    return area;
}

things_and_stuff:
    sub $sp, $sp, 8
    sw $ra, 0($sp)
    # s is on offsets 4-7 of the stack
    la $a0, 4($sp) # &s
class animal {
public:
    animal (unsigned age) : age_(age) {
        // remember to set up vtable pointer
    }

    virtual void speak() = 0;

    virtual unsigned age() {
        return age_;  
    }

protected:
    unsigned age_;  
};

class fox : public animal {
public:
    fox(unsigned age) : animal(age) {
        // remember to set up vtable pointer
    }

    virtual void speak() {
        puts("Ring-ding-ding-ding-dingeringeding!");
    }
};

unsigned speak_and_get_age(animal *a) {
    a->speak();
    return a->age();
}

unsigned get_min_age() {
    fox f(16);
    unsigned age = speak_and_get_age(&f);
    return age / 2 + 7;
}
class animal {
public:
    animal (unsigned age) : age_(age) {
        // remember to set up vtable pointer
    }

    virtual void speak() = 0;

    virtual unsigned age() {
        return age_;  
    }

protected:
    unsigned age_;  
};

.data
animal_vtable:
    # pure virtual methods get a NULL (0) entry

.text
animal_animal:

animal_age:
class fox : public animal {
public:
    fox(unsigned age) : animal(age) {
        // remember to set up vtable pointer
    }

    virtual void speak() {
        puts("Ring-ding-ding-ding-dingeringeding!");
    }
};

.data
what_the_fox_say:
    .asciiz "Ring-ding-ding-ding-dingeringeding!"

fox_vtable:

.text
fox_fox:
    sub $sp, $sp, 8
    sw $ra, 0($sp)
    sw $a0, 4($sp)

    jal animal_animal
    lw $a0, 4($sp)
    la $t0, _______________
    sw $t0, ___($a0)

    lw $ra, 0($sp)
    add $sp, $sp, 8
    jr $ra

fox_speak:
unsigned speak_and_get_age(animal *a) {
    a->speak();
    return a->age();
}
speak_and_get_age:
unsigned get_min_age() {
    fox f(16);
    unsigned age = speak_and_get_age(&f);
    return age / 2 + 7;
}

get_min_age:
    sub $sp, $sp, 12
    sw $ra, 0($sp)
    # f is on offsets 4-11 of the stack
    div $v0, $v0, 2
    add $v0, $v0, 7
    lw $ra, 0($sp)
    add $sp, $sp, 12
    jr $ra