
SOURCE CODE EXAMPLES: C++

This appendix contains the code examples referred to in the main text. There is a version of the appendix for each language. This is the C++ version.

#1 EXAMPLE FOR THE CALLING DIAGRAM

```
int main() {
    function_one();
    function_two();
}

int function_one() {
    function_three();
}

int function_two() {
    function_four();
    function_one();
    function_two();
}

int function_three() {}

int function_four() {
    function_three();
}
```

#2 SMALL EXAMPLE FOR HAND SIMLUATION

```
char v1 = 10;
int v2 = v1 * 100;

while (v2 > 500) {
    cout << "Excess over 500 = " << v2 - 500;
    v2 = --v1 * 100;
}
```

#3 EXAMPLE FOR SCOPE REGIONS

```
void A() {
    int B;
}

void E() {
    int C;
    {
        int D;
    }
}
```

#4 LARGE EXAMPLE FOR HAND SIMULATION

```
#include <stdio.h>

int g1 = 100, g2 = 20;

void f1(int xx) {
    int w = 10;
    w = w + g1;
    printf("%d", w / xx);
}

int f2() {
    int w = 5;
    g1 = g2 + w;
    return w + 2;
}

int main() {
    int r;
    r = f2();
    f1(r);
    printf("%d", g1);
    return 0;
}
```

#5 TOP-DOWN DESIGN EXAMPLE

```
/******  
The paint estimator  
******/  
  
#include <stdio.h>  
const float paintCost = 12.5;  
const float height = 8.0;  
  
float costOfPainting, length, width,  
      coverage, area; /* global variables */  
  
void inputValues() {  
    printf("Type in length, width, coverage: ");  
    scanf("%f%f%f", &length, &width, &coverage);  
}  
  
void calculateArea() {  
    area = 2 * (length + width) * height + length * width;  
}  
  
void calculateCostFromArea() {  
    costOfPainting = area / coverage * paintCost;  
}  
  
void calculateCost() {  
    calculateArea();  
    calculateCostFromArea();  
}  
void displayCost() {  
    printf("Cost of painting a room is %f\n",  
          costOfPainting);  
}  
int main() {  
    inputValues();  
    calculateCost();  
    displayCost();  
}
```

#6 TOP-DOWN DESIGN EXAMPLE, REWORKED

```
/******  
The paint estimator  
*****/  
  
#include <stdio.h>  
const float paintCost = 12.5;  
const float height = 8.0;  
  
float costOfPainting, length, width, coverage;  
  
void inputValues() {  
    printf("Type in length, width, coverage: ");  
    scanf("%f%f%f", &length, &width, &coverage);  
}  
float calculateArea() {  
    return 2 * (length + width) * height + length * width;  
}  
  
float calculateCostFromArea() {  
    return area / coverage * paintCost;  
}  
  
void calculateCost(float l, float w, float c) {  
    float area;  
  
    area = calculateArea(l, w);  
    calculateCostFromArea(area, c);  
}  
void displayCost(float l, float w, float c, float cp) {  
    printf("Cost of painting a room %f by %f with\n  
        paint of coverage %f is %f\n",  
        l, w, c, cp);  
}  
  
int main() {  
    float costOfPainting;  
  
    inputValues();  
    costOfPainting = calculateCost(length, width, coverage);  
    displayCost(length, width, coverage, costOfPainting);  
}
```

#7 THE MAIN PROGRAM FOR THE TOP-DOWN DESIGN EXAMPLE

```
int main() {
    float costOfPainting, length, width, coverage;

    length = inputLength();
    width = inputWidth();
    coverage = inputCOverage();
    calculateCost(length, width, coverage);
    displayCost(length, width, coverage, costOfPainting);
}
```

#8 THE PAINT ESTIMATOR DESIGN IN CLASSES

```
class Room {
private:
    float Width, Length; // in ft.
    const float Height; // in ft.
};

class Paint {
private:
    const float CostOfPaint; // in $/gallon
    float Coverage; // in sq.ft./gallon
};

class Estimator {
private:
    Room room;
    Paint paint;
    float CostOfPainting; // in $
};
```

#9 THE PAINT ESTIMATOR DESIGN WITH CLASS METHODS

```
class Room {
private:
    float Width, Length; // in ft.
    const float Height; // in ft.
public:
    Room() : Height(8.0) {} // a constructor to set the constant
    float getWidth() { return Width; } // accessor method
    float getLength() { return Length; } // accessor method
    float getHeight() { return Height; } // accessor method
    void input() {
        cout << "Type width and length: "; // input method prompts
        cin >> Width >> Height; // the user
    }
};
```

```

class Paint {
private:
    const float CostOfPaint; // in $/gallon
    float Coverage;          // in sq.ft./gallon
public:
    Paint() : CostOfPaint(12.5) {} // a constructor to set the
                                   // constant

    float getCoverage() { return Coverage; } // accessor method
    float getCostOfPaint() { return CostOfPaint; } // accessor method
    void input() { // input method
        cout << "Type coverage: ";
        cin >> Coverage;
    }
};

class Estimator {
private:
    Room room;
    Paint paint;
    float CostOfPainting; // in $
public:
    Estimator() {} // default constructor
    void estimate();
    void display() { cout << "Cost is " << CostOfPainting; }
    void run();
};

```

#10 THE PAINT ESTIMATOR TOP LEVEL METHODS

```

void Estimator::estimate() {
    float wallArea =
        2 * (room.getWidth() + room.getLength()) * room.getHeight();
    float ceilingArea = room.getWidth() * room.getLength();
    CostOfPainting = (wallArea + ceilingArea) / paint.getCoverage() *
        paint.getCost();
}

void Estimator::run() {
    room.input();
    paint.input();
    estimate();
    display();
}

int main() {
    Estimator estimator;
    estimator.run();
}

```

#11 SIMPLE EXAMPLE FOR DEBUGGING

```

while (num != 9999) {
    total += num;
    scanf("%d", &num);
}

```

#12 DEBUGGING WITH TRACE ADDED

```
while (num != 9999) {
    printf("Trace total: %d\n", total);
    total += num;
    scanf("%d", &num);
}
```

#13 ADDING AN ASSERTION

Add a line with `#include <assert.h>` at the top of the file.

```
assert(i >= 0 && i <= 9);
```

#14 AN ASSERTION IN STRAIGHT CODE

```
if (x < 0 || x > 9) {
    printf("Error: x is out of range (%d)\n", x);
    exit(1);
}
```

#15 A DEBUGGING EXAMPLE

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int c, lastc, totalIEs = 0, totalEIs = 0;

    printf("Type any number of lines followed by EOF\n");

    while ((c = getchar()) != EOF) {
        c = tolower(c);
        if (c == 'i' && lastc == 'e')
            ++totalEIs;
        else if (c == 'e' && lastc == 'i')
            ++totalIEs;
        else
            lastc = c;
    }
    printf("\nthe number of IEs is %d\n", totalIEs);
    printf("\nand the number of EIs = %d\n", totalEIs);
}
```

#16 THE EXAMPLE WITH TRACE ADDED

```

int main() {
    int c, lastc, totalIEs = 0, totalEIs = 0;

    printf("Type any number of lines followed by EOF\n");

    while ((c = getchar()) != EOF) {
        c = tolower(c);
        if (c == 'i' && lastc == 'e')
            ++totalEIs;
        else if (c == 'e' && lastc == 'i')
            ++totalIEs;
        else
            lastc = c;
        printf(
            "c = %c lastc = %c totalIEs = %d totalEIS = %d\n",
            c, lastc, totalIEs, totalEIs);
    }

```

#17 THE EXAMPLE WITH ASSERTION ADDED

```

else
    lastc = c;
    /*****
    assert(lastc == c);
    *****/
}
...

```

#18 THE EXAMPLE ASSERTION AND TRACE

```

...
else
    lastc = c;
    /*///// a trace statement      /////*/
    printf("lastc = %c, c = %c\n", lastc, c);
    /*////////////////////////////////////*/
    /****** an assertion *****/
    /* assert(lastc == c);          */
    /*////////////////////////////////////*/
}

```

...

#19 THE EXAMPLE DEBUGGED

```
...
#include <stdio.h>
#include <stdlib.h>

int main() {
    int c, lastc, totalIEs = 0, totalEIs = 0;

    printf("Type any number of lines followed by EOF\n");

    while ((c = getchar()) != EOF) {
        c = tolower(c);
        if (c == 'i' && lastc == 'e')
            ++totalEIs;
        else if (c == 'e' && lastc == 'i')
            ++totalIEs;
        /*else*/
        lastc = c;
    }
}
```

