

# ECE 2400 Computer Systems Programming

## Topic 4: C Pointers

<http://www.cs1.cornell.edu/courses/ece2400>  
School of Electrical and Computer Engineering  
Cornell University

revision: 2026-02-09-11-31

Please do not ask for solutions. Students should compare their solutions to solutions from their fellow students, discuss their solutions with the instructors during lab/office hours, and/or post their solutions on Ed for discussion.

### List of Problems

<b>1 Short Answer</b>	<b>2</b>
1.A Working with Lines . . . . .	2
1.B Conceptual Storage vs. Machine Memory for Pointers . . . . .	3

**Problem 1. Short Answer**

Carefully plan your solution before starting to write your response. Please be brief and to the point; if at all possible, limit your answers to the space provided.

**Part 1.A Working with Lines**

The following `calc_slope` function calculates the slope of a 2D line, and the `translate_x` function translates a 2D line in the X direction by a given shift amount. **Draw the stack diagram that corresponds to the execution of this C program.** You must clearly label all variables in your diagram.

```

000000 01 // User-defined types for points and lines
000000 02
000000 03 typedef struct { int x; int y; } point_t;
000000 04 typedef struct { point_t pt0; point_t pt1; } line_t;
000000 05
000000 06 // Function for calculating the slope
000000 07
000000 08 float calc_slope( line_t line )
000000 09 {
000000 10     float rise = line.pt1.y - line.pt0.y;
000000 11     float run  = line.pt1.x - line.pt0.x;
000000 12     return rise / run;
000000 13 }
000000 14
000000 15 // Function for translating a line
000000 16
000000 17 void translate_x( line_t* line_p, int shift )
000000 18 {
000000 19     line_p->pt0.x += shift;
000000 20     line_p->pt1.x += shift;
000000 21 }
000000 22
000000 23 // Main function
000000 24
000000 25 int main( void )
000000 26 {
000000 27     line_t line;
000000 28     line.pt0.x = 1;
000000 29     line.pt0.y = 1;
000000 30     line.pt1.x = 2;
000000 31     line.pt1.y = 3;
000000 32
000000 33     float slope = calc_slope( line );
000000 34     translate_x( &line, 4 );
000000 35
000000 36     return 0;
000000 37 }

```

stack

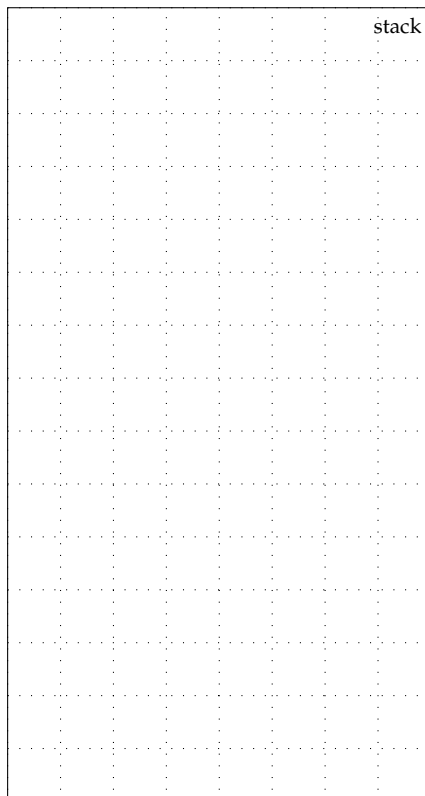
**Part 1.B Conceptual Storage vs. Machine Memory for Pointers**

The following code snippet illustrates using pointers. **Draw the conceptual stack diagram that corresponds to the execution of this C program.** Clearly label all variables.

Once you have finished the conceptual stack diagram, **draw the machine memory diagram that also corresponds to the execution of this C program.** Assume that the machine only has a total of 128 bytes of memory. Clearly label the location of each variable in memory. We have already allocated the variable a to get you started. Recall that a variable of type int is 32 bits (four bytes), and that we always arrange variables such that the least significant (“right most”) byte is at the lowest address in memory. Assume that pointers are also 32 bits (four bytes). You do not need to show values in machine memory in base two. Each byte can be a decimal number. You do not need to show how code maps into machine memory.

```

0001 int    a    = 42;
0002 int    b    = 13;
0003 int*   ptr0 = &a;
0004 int*   ptr1 = ptr0;
0005 int**  ptr2 = &ptr1;
0006 int    c    = *ptr0 + **ptr2;
    
```



Memory (byte addr)

