Calvin Liu

Homework 3

3.59, 3.64, 3.67, 3.70

3.59.

int switch\_prob(int x, int n)

{

int result = x;

switch(n) {

case 50:

case 54:

result <<= 2;

break;

case 55:

result >>= 2;

break;

case 56:

result \*= 3;

case 57:

result \*= result;

default:

result += 10;

}

return result;

}

3.64.

1. We can see in lines 5–7 of the code for word\_sum that it appears as if three values are being retrieved from the stack, even though the function has only a single argument. Describe what these three values are.

* Line 6 is for int a
* Line 7 is for int \*p
* Line 5 is for the pointer for struct 2

B. We can see in line 4 of the code for prod that 20 bytes are allocated in the stack frame. These get used as five fields of 4 bytes each. Describe how each of these fields gets used.

- The ret command

- int x

- int y

- s1.a

- s1.p

C. How would you describe the general strategy for passing structures as argu- ments to a function?

The structure will be passed onto the stack via pointer and if you want the contents, you have to dereference it.

D. How would you describe the general strategy for handling a structure as a return value from a function?

You must pop the contents of the structure before returning.

3.67.

1. What would be the offsets (in bytes) of the following fields: e1.p:  e1.y: e2.x: e2.next:

* e1.p – 4 bytes
* e1.y – 4 bytes
* e2.x – 4 bytes
* e2.next – 4 bytes

1. How many total bytes would the structure require?

The total bytes would be 12 bytes

1. The compiler generates the following assembly code for the body of proc: On the basis of this information, fill in the missing expressions in the code for proc. **Hint:** Some union references can have ambiguous interpretations. These ambiguities get resolved as you see where the references lead. There is only one answer that does not perform any casting and does not violate any type constraints.

void proc (union ele \*up)

{

up-> e2.next->e1.y= \*(up-> e2.next->e1.p) - up-> e2.x;

}

3.70.

A. Generate a C version of the function.

long traverse(tree\_ptr tp)

{

if(tp == NULL)

{

return tp->val;

}

int x = traverse(tp->right);

int y = traverse(tp->left);

if(x > y)

{  
 return x;

}

else

return y;

}

 B. Explain in English what this function computes.

It traverses the tree and computes the largest value