# Gimple Code

;; Function main (main, funcdef\_no=0, decl\_uid=1794, cgraph\_uid=0, symbol\_order=0)

main ()

{

int d;

int c;

int b;

int a;

int D.1804;

int \_10;

<bb 2> [0.00%]:

d\_3 = 10;

if (c\_4(D) > d\_3)

goto <bb 3>; [0.00%]

else

goto <bb 4>; [0.00%]

<bb 3> [0.00%]:

a\_7 = 3;

b\_8 = 2;

goto <bb 5>; [0.00%]

<bb 4> [0.00%]:

a\_5 = 2;

b\_6 = 3;

<bb 5> [0.00%]:

# a\_1 = PHI <a\_7(3), a\_5(4)>

# b\_2 = PHI <b\_8(3), b\_6(4)>

c\_9 = a\_1 + b\_2;

\_10 = 0;

<L3> [0.00%]:

return \_10;

}

# C Code

int main()

{

int a, b, c, d;

d = 10;

if (c > d) {

a = 3;

b = 2;

} else {

a = 2;

b = 3;

c = a + b;

}

# Answers

1] Local variables are suffixed with a unique number which represents the assignment. For example, d\_3 refers to the usage of variable d. Variables that are likely to be used before any assignment are suffixed with D to indicate that the assignment was implicit in the declaration. For example, variable c is used as c\_4(D) in the `if' condition.

2] A PHI node merges assignments from different paths and creates a new assignment. For example:

# a\_1 = PHI <a\_5(3), a\_7(4)>

indicates that if the control reaches here from basic block 3 then use the value of a\_5; otherwise if the control reaches here from basic block 4 then use the value of a\_7.

3] int main()

{

int a, b;

switch (a)

{

case 1:

b = 2;

break;

case 2:

b = 3;

break;

case 3:

b = 4;

case 4:

b = 5;

break;

case 5:

b = 6;

break;

default:

b = 7;

}

return b;

}

In this case the PHI node at the end of the switch statement will merge multiple assignments such as:

#b\_1=PHI<b\_4(3),b\_5(4),b\_7(5),b\_8(6),b\_9(7)>