POLITECNICO
MILANO 1863

Software Engineering 2

V&V Exercises

M Camilli, E Di Nitto, M Rossi

Verification & Validation

Exercises: Static Analysis, Symbolic Execution

Exercise 1

- Consider the function `foo`, written in a C-like language:

- Execute `foo` symbolically limiting the execution of the loop statement to exactly 2 iterations. Show, for each non-conditional statement:
  - <path condition, symbolic state>
- Define the pre-condition to the execution of `foo` s.t. the while loop is executed exactly twice
- Generate 3 possible test cases to run this path

```
int foo(int a, int b) {
  a++;
  while (a < b) {
    if (a != b)
      a++;
  }
  return a;
}
```

Exercise 1

- Limiting the execution of the loop to exactly 2 iterations.

```
0: int foo(int a, int b) {
1:   a++;
2:   while (a < b) {
3:     if (a != b)
4:       a++;
5:   }
6:   return a;
7: }
```

<0>

a	b	π
A	B	T

<0,1>

a	b	π
A+1	B	T

<0,1,2>

a	b	π
A+1	B	A+1 < B

<0,1,2,3>

a	b	π
A+1	B	A+1 < B A+1 ≠ B

<0,1,2,3,4>

a	b	π
A+2	B	A+1 < B

<0,1,2,3,4,2>

a	b	π
A+2	B	A+2 < B

<0,1,2,3,4,2,3>

a	b	π
A+2	B	A+2 < B A+2 ≠ B

<0,1,2,3,4,2,3,4>

a	b	π
A+3	B	A+2 < B

<0,1,2,3,4,2,3,4,2>

a	b	π
A+3	B	A+2 < B A+3 ≥ B

<0,1,2,3,4,2,3,4,2,6>

a	b	π
A+3	B	A+3 = B

Exercise 1

- Precondition to execute foo s.t. the loop is executed exactly 2 times

```
0: int foo(int a, int b) {
1:   a++;
2:   while (a < b) {
3:     if (a != b)
4:       a++;
5:   }
6:   return a;
7: }
```

<0,1,2,3,4,2,3,4,2,6>

a	b	π
A+3	B	A+3 = B

SAT ✓

=> precondition: { b = a+ 3 }

- Three possible test cases
  - {a = 1, b = 4}, {a = 0, b = 3}, {a = -3, b = 0}

Exercise 2

- Consider the following function, written in a C-like language:

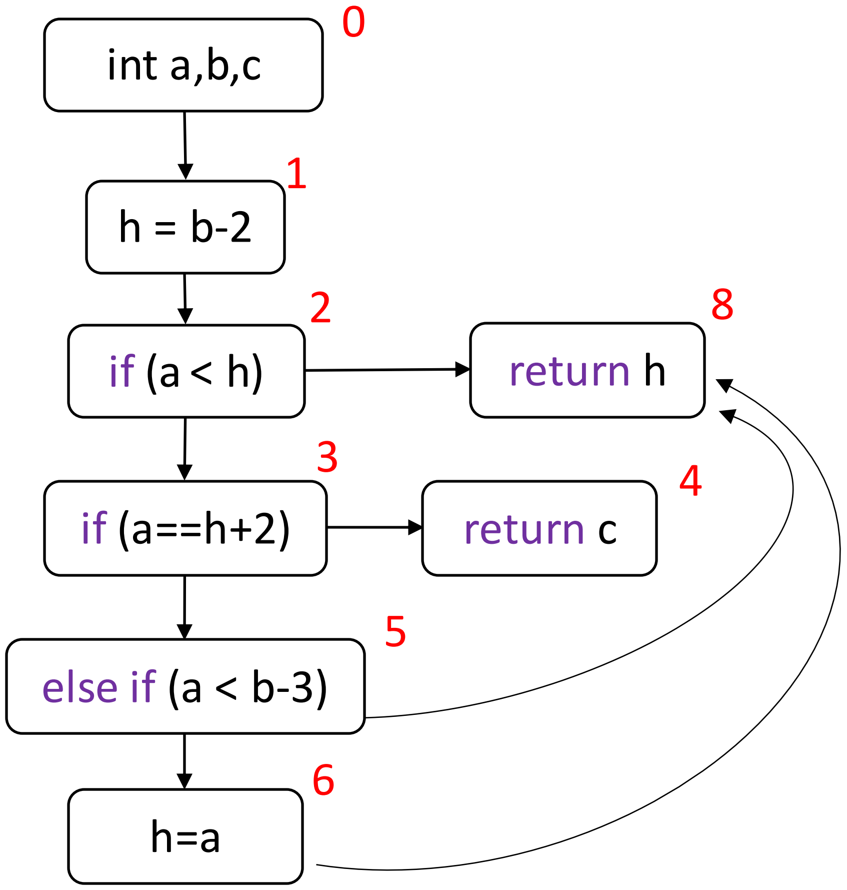
```
0:  int bar(int a, int b, int c) {
1:    int h = b-2;
2:    if (a < h) {
3:      if (a == h+2)
4:        return c;
5:      else if (a < b-3)
6:        h = a;
7:   }
8:   return h;
9: }
```

- Derive the CFG
- Derive the set of live variables at the exit of each block. Are there dead variables after definition at block 0?
- Use symbolic execution to explore all paths in the function

Exercise 2

- CFG structure

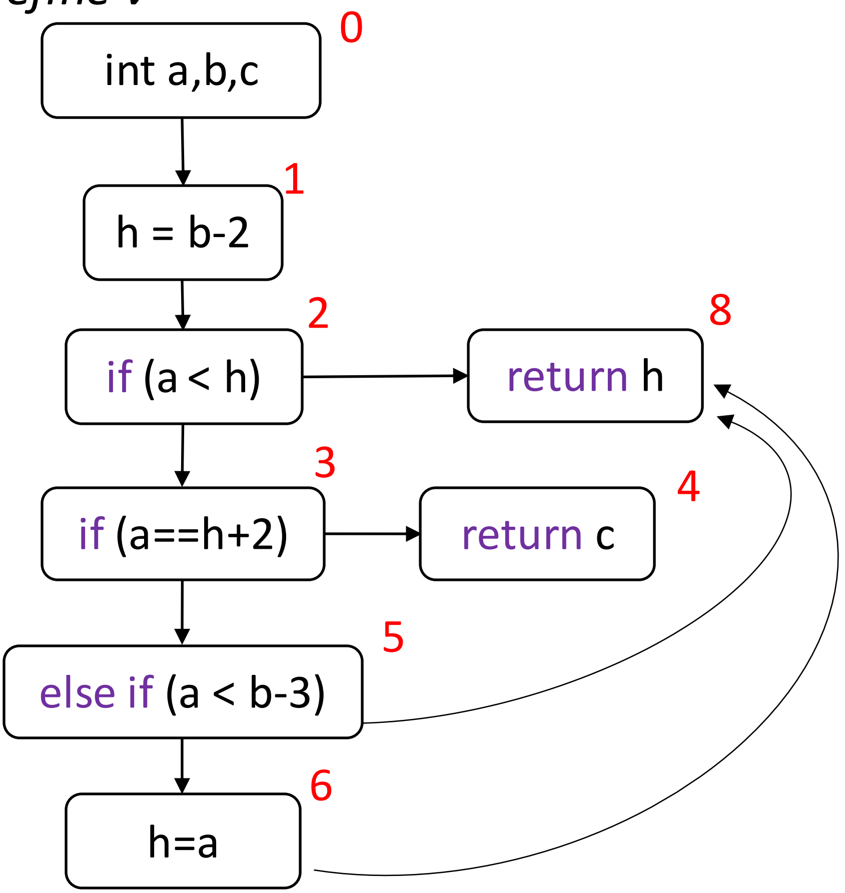
```
0:  int bar(int a, int b, int c) {
1:    int h = b-2;
2:    if (a < h) {
3:      if (a == h+2)
4:        return c;
5:      else if (a < b-3)
6:        h = a;
7:   }
8:   return h;
9: }
```



Exercise 2

- Live variables: *Given a CFG, a variable v is live at the exit of a block b if there is some path (on the CFG) from block b to a use of v that does not redefine v*

LV(0) = {a,b,c}
LV(1) = {a,b,c,h}
LV(2) = {a,b,c,h}
LV(3) = {a,b,c,h}
LV(4) = { }
LV(5) = {a,h}
LV(6) = {h}
LV(8) = { }



- All variables defined at block 0 may be live after 0

Exercise 2

- Symbolic execution path <0,1,2,3,4>

```
0:  int bar(int a, int b, int c) {
1:    int h = b-2;
2:    if (a < h) {
3:      if (a == h+2)
4:        return c;
5:      else if (a < b-3)
6:        h = a;
7:    }
8:    return h;
9: }
```

<0>

a	b	c	π
A	B	C	T

<0,1>

a	b	c	h	π
A	B	C	B-2	T

<0,1,2>

a	b	c	h	π
A	B	C	B-2	A<B-2

<0,1,2,3,4>

UNSAT ✗

a	b	c	h	π
A	B	C	B-2	A<B-2 A=B-2+2

Path <0,1,2,3,4> is the only one where c is used (with no redefinition) after definition at block 0, so, c is actually dead

Exercise 2

- Symbolic execution
  path <0,1,2,3,5,6,8>

```
0:  int bar(int a, int b, int c) {
1:    int h = b-2;
2:    if (a < h) {
3:      if (a == h+2)
4:        return c;
5:      else if (a < b-3)
6:        h = a;
7:    }
8:   return h;
9: }
```

<0,1,2>

a	b	c	h	π
A	B	C	B-2	A<B-2

<0,1,2,3>

a	b	c	h	π
A	B	C	B-2	A<B-2 A≠B-2+2

<0,1,2,3,5>

a	b	c	h	π
A	B	C	B-2	A<B-2 A≠B A<B-3

It can be simplified

<0,1,2,3,5,6,8>

a	b	c	h	π	SAT ✓
A	B	C	A	A<B-3	

Exercise 2

- Symbolic execution path <0,1,2,3,5,8>

```
0:   int bar(int a, int b, int c) {
1:     int h = b-2;
2:     if (a < h) {
3:       if (a == h+2)
4:         return c;
5:       else if (a < b-3)
6:         h = a;
7:    }
8:    return h;
9: }
```

<0,1,2>

a	b	c	h	π
A	B	C	B-2	A<B-2

<0,1,2,3>

a	b	c	h	π
A	B	C	B-2	A<B-2 A≠B-2+2

<0,1,2,3,5>

a	b	c	h	π
A	B	C	B-2	A<B-2 A≠B A≥B-3

<0,1,2,3,5,8>

a	b	c	h	π	SAT ✓
A	B	C	B-2	A=B-3	

- Symbolic execution path <0,1,2,8>

<0,1,2>

a	b	c	h	π
A	B	C	B-2	A≥B-2

<0,1,2,8>

a	b	c	h	π	SAT ✓
A	B	C	B-2	A≥B-2	

Exercise 3

- Consider the following function, written in a C-like language, where `rand()` returns a pseudo-random (integer) number:

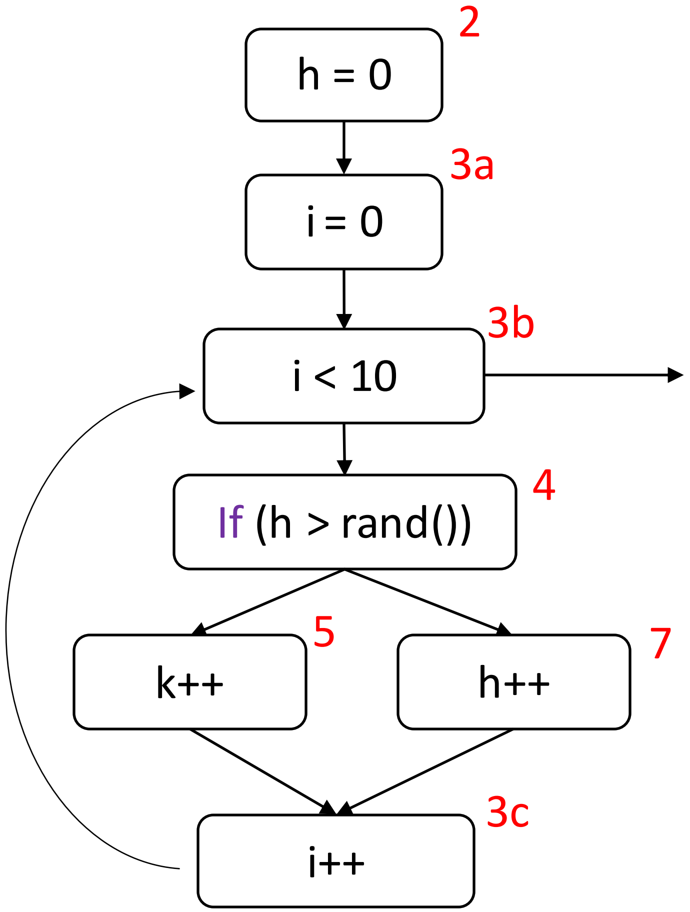
```
void foo( ) {
  int h, k;
  h = 0;
  for (int i=0; i<10; i++) {
    if (h > rand())
      k++;
    else
      h++;
  }
}
```

- Build the CFG of `foo`
- Derive all the reaching definitions at the entry and the exit of each block
- According to the reaching definitions, derive all the UD chains and then def-use pairs for variables h and k
- According to the previous results, what are the potential problems of `foo`?

Exercise 3

- CFG structure

```
0:   void foo( ) {
1:     int h, k;
2:     h = 0;
3:     for (int i=0; i<10; i++) {
4:       if (h > rand())
5:         k++;
6:       else
7:         h++;
8:     }
9:   }
```



Exercise 3

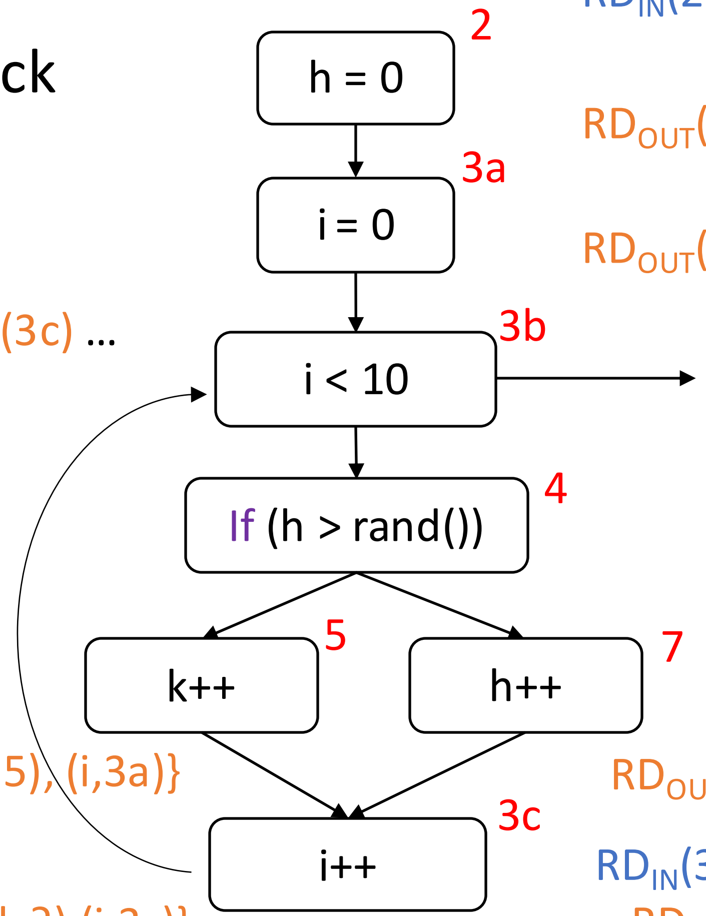
- RD entry/exit of each block

RD$_{IN}$(2) = { (h,?), (k,?), (i,?)}

RD$_{OUT}$(2) = {(h,2), (k,?), (i,?)} = RD$_{IN}$(3a)

RD$_{OUT}$(3a) = {(h,2), (k,?), (i,3a)}

RD$_{IN}$(3b) = RD$_{OUT}$(3a) U RD$_{OUT}$(3c) …
TBD



RD$_{OUT}$(3b) = {(h,2), (k,?), (i,3a)} = RD$_{IN}$(4)

RD$_{OUT}$(4) = {(h,2), (k,?), (i,3a)}
= RD$_{IN}$(5) = RD$_{IN}$(7)

RD$_{OUT}$(5) = {(h,2), (k,5), (i,3a)}

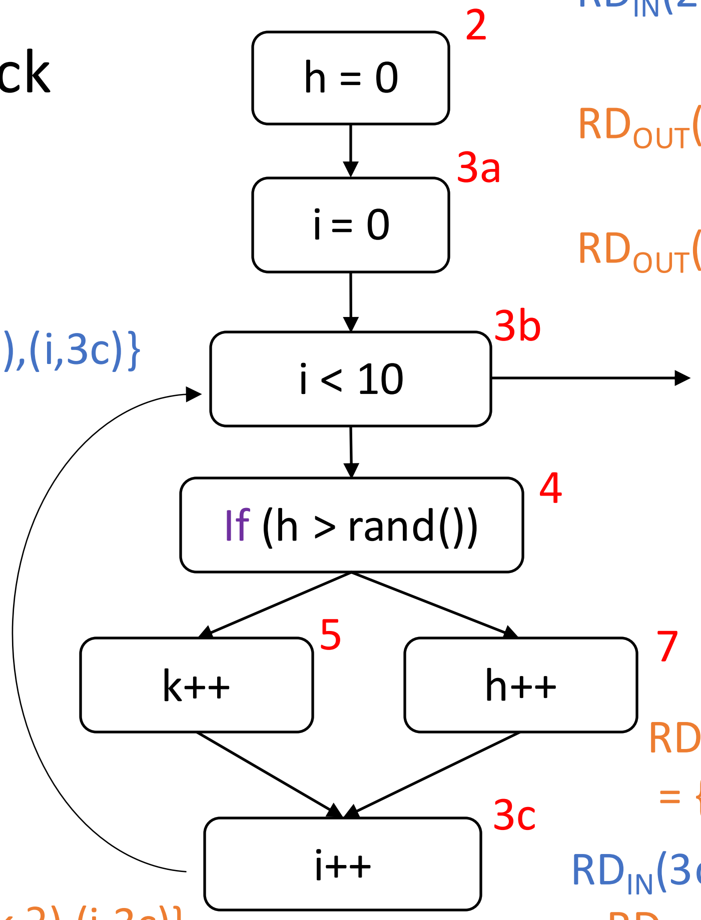
RD$_{OUT}$(7) = {(h,7), (k,?), (i,3a)}

RD$_{IN}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3a)}
= RD$_{OUT}$(5) U RD$_{OUT}$(7)

RD$_{OUT}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3c)}

Exercise 3

- RD entry/exit of each block



$RD_{IN}(2)$ = { (h,?), (k,?), (i,?)}

$RD_{OUT}(2)$ = {(h,2), (k,?), (i,?)} = $RD_{IN}(3a)$

$RD_{OUT}(3a)$ = {(h,2), (k,?), (i,3a)}

$RD_{IN}(3b)$ = {(h,2),(k,?),(i,3a),(h,7),(k,5),(i,3c)}

$RD_{OUT}(3b)$ = ~~{(h,2), (k,?), (i,3a)}~~ =
{(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = $RD_{IN}(4)$

$RD_{OUT}(4)$ = ~~{(h,2), (k,?), (i,3a)}~~ =
{(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = $RD_{IN}(5)$ =
$RD_{IN}(7)$

$RD_{OUT}(7)$ ~~= {(h,7), (k,?), (i,3a)}~~ =
= {(h,7),(k,?),(k,5),(i,3a),(i,3c)}

$RD_{OUT}(5)$ ~~= {(h,2), (k,5), (i,3a)}~~ =
= {(h,2),(h,7),(k,5),(i,3a),(i,3c)}

$RD_{IN}(3c)$ = {(h,2),(h,7),(k,5),(k,?),(i,3a),(i,3c)}
= $RD_{OUT}(5)$ U $RD_{OUT}(7)$

$RD_{OUT}(3c)$ = {(h,2),(h,7),(k,5),(k,?),(i,3c)}

Exercise 3

- UD chains

RD$_{\text{IN}}$(2) = { (h,?), (k,?), (i,?)}

RD$_{\text{OUT}}$(2) = {(h,2), (k,?), (i,?)} = RD$_{\text{IN}}$(3a)

RD$_{\text{OUT}}$(3a) = {(h,2), (k,?), (i,3a)}

RD$_{\text{IN}}$(3b) = (h,2),(k,?),(i,3a),(h,7),(k,5),(i,3c)}

RD$_{\text{OUT}}$(3b) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD$_{\text{IN}}$(4)

RD$_{\text{OUT}}$(4) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD$_{\text{IN}}$(5) = RD$_{\text{IN}}$(7)

RD$_{\text{OUT}}$(5) = {(h,2),(h,7),(k,5),(i,3a),(i,3c)}

RD$_{\text{OUT}}$(7) = {(h,7),(k,?),(k,5),(i,3a),(i,3c)}

RD$_{\text{IN}}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3a),(i,3c)}

RD$_{\text{OUT}}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3c)}

- UD(h,4) = {2,7}
- UD(k,5) = {5,?}
- UD(h,7) = {2,7}

Exercise 3

- Def-use pairs

RD$_{IN}$(2) = { (h,?), (k,?), (i,?)}

RD$_{OUT}$(2) = {(h,2), (k,?), (i,?)} = RD$_{IN}$(3a)

RD$_{OUT}$(3a) = {(h,2), (k,?), (i,3a)}

RD$_{IN}$(3b) = (h,2),(k,?),(i,3a),(h,7),(k,5),(i,3c)}

RD$_{OUT}$(3b) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD$_{IN}$(4)

RD$_{OUT}$(4) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD$_{IN}$(5) = RD$_{IN}$(7)

RD$_{OUT}$(5) = {(h,2),(h,7),(k,5),(i,3a),(i,3c)}

RD$_{OUT}$(7) = {(h,7),(k,?),(k,5),(i,3a),(i,3c)}

RD$_{IN}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3a),(i,3c)}

RD$_{OUT}$(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3c)}

- UD(h,4) = {2,7}
- UD(k,5) = {5,?}
- UD(h,7) = {2,7}
- Def-use pairs
  - h: <2,4>, <7,4>, <2,7>, <7,7>
  - k: <5,5>, <?,5>

Exercise 3

- Possible issues

RD_{IN}(2) = { (h,?), (k,?), (i,?)}

RD_{OUT}(2) = {(h,2), (k,?), (i,?)} = RD_{IN}(3a)

RD_{OUT}(3a) = {(h,2), (k,?), (i,3a)}

RD_{IN}(3b) = (h,2),(k,?),(i,3a),(h,7),(k,5),(i,3c)}

RD_{OUT}(3b) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD_{IN}(4)

RD_{OUT}(4) = {(h,2),(h,7),(k,?),(k,5),(i,3a),(i,3c)} = RD_{IN}(5) = RD_{IN}(7)

RD_{OUT}(5) = {(h,2),(h,7),(k,5),(i,3a),(i,3c)}

RD_{OUT}(7) = {(h,7),(k,?),(k,5),(i,3a),(i,3c)}

RD_{IN}(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3a),(i,3c)}

RD_{OUT}(3c) = {(h,2),(h,7),(k,5),(k,?),(i,3c)}

- UD(h,4) = {2,7}
- UD(k,5) = {5,?}
- UD(h,7) = {2,7}
- Def-use pairs
  - h: <2,4>, <7,4>, <2,7>, <7,7>
  - k: <5,5>, <?,5> → possible use without definition

Exercise 4

- Consider the following function, written in a C-like language:

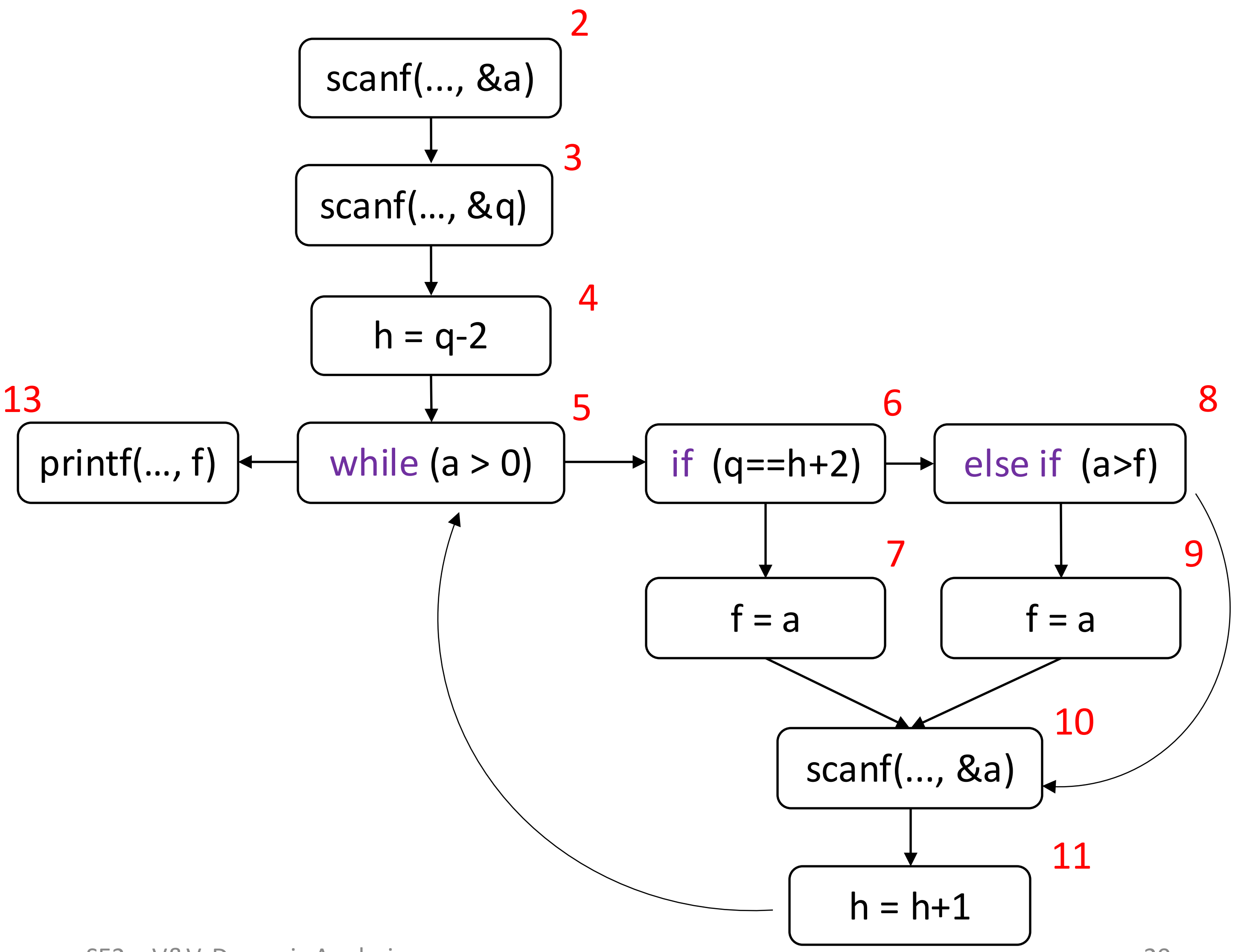
```
void main() {
  int a, h, f, q;
  scanf(“%d”, &a);
  scanf(“%d”, &q);
  h = q-2;
  while (a > 0) {
      if (q == h+2)
        f = a;
      else if (a > f)
        f = a;
      scanf(“%d”, &a);
      h = h+1;
  }
  printf(“%d”, f);
}
```

- Derive the CFG
- Derive the reaching definitions, the UD chains, and def-use pairs for all variables
- Explain potential issues (if any) highlighted by the def-use analysis
- Use symbolic execution to show whether the potential problems (def-use analysis) can occur or not

Exercise 4

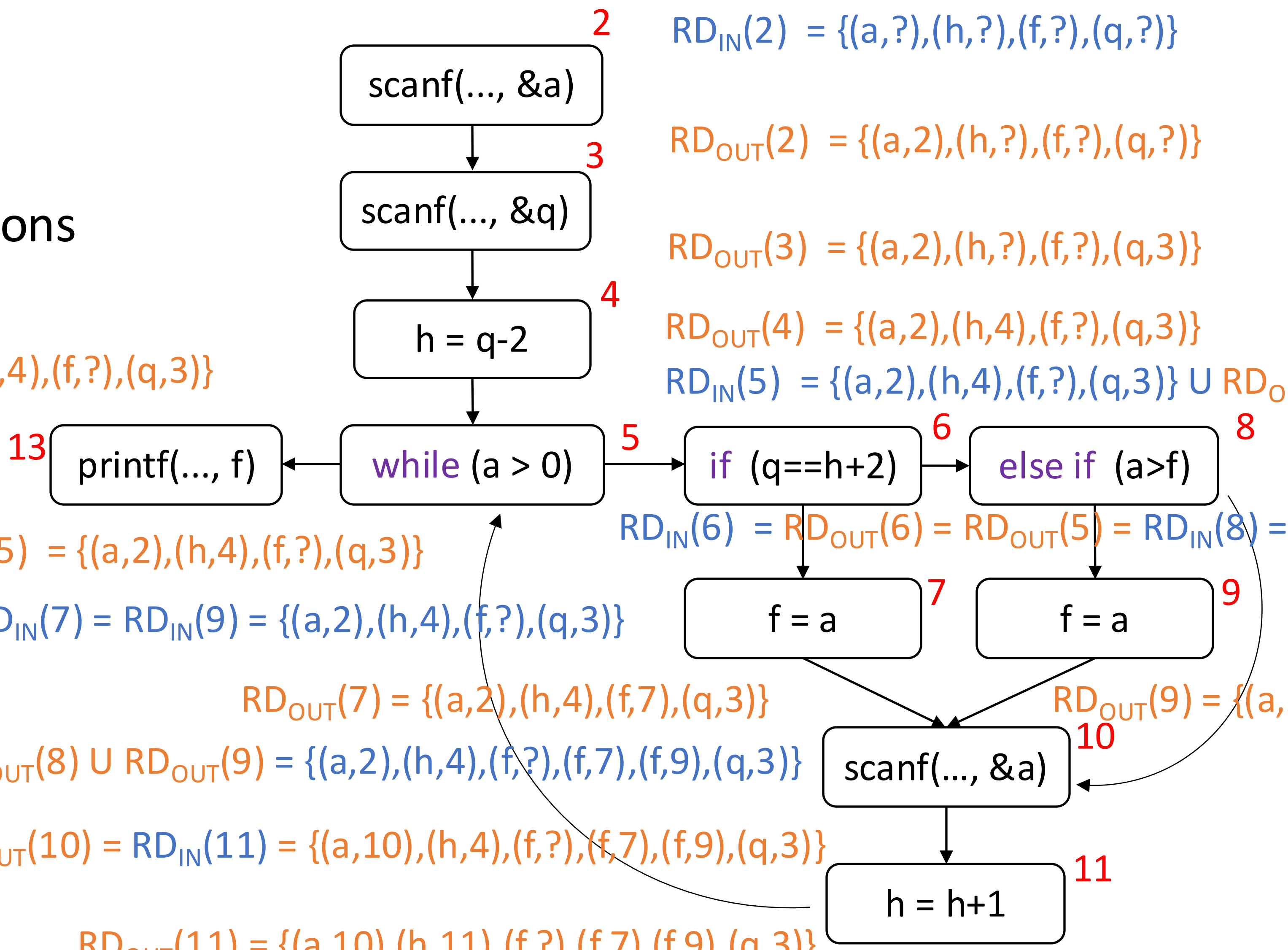
- CFG

```
0:  void main() {
1:    int a, h, f, q;
2:    scanf(“%d”, &a);
3:    scanf(“%d”, &q);
4:    h = q-2;
5:    while (a > 0) {
6:        if (q == h+2)
7:          f = a;
8:        else if (a > f)
9:          f = a;
10:       scanf(“%d”, &a);
11:       h = h+1;
12:   }
13:   printf(“%d”, f);
14: }
```



Exercise 4

- Reaching definitions



$RD_{IN}(2) = \{(a,?),(h,?),(f,?),(q,?)\}$

$RD_{OUT}(2) = \{(a,2),(h,?),(f,?),(q,?)\}$

$RD_{OUT}(3) = \{(a,2),(h,?),(f,?),(q,3)\}$

$RD_{OUT}(4) = \{(a,2),(h,4),(f,?),(q,3)\}$

$RD_{IN}(5) = \{(a,2),(h,4),(f,?),(q,3)\} \cup RD_{OUT}(11)$ ... TBD

$RD_{OUT}(13) = \{(a,2),(h,4),(f,?),(q,3)\}$

$RD_{IN}(6) = RD_{OUT}(6) = RD_{OUT}(5) = RD_{IN}(8) = RD_{OUT}(8)$

$RD_{OUT}(5) = \{(a,2),(h,4),(f,?),(q,3)\}$

$RD_{IN}(7) = RD_{IN}(9) = \{(a,2),(h,4),(f,?),(q,3)\}$

$RD_{OUT}(7) = \{(a,2),(h,4),(f,7),(q,3)\}$

$RD_{OUT}(9) = \{(a,2),(h,4),(f,9),(q,3)\}$

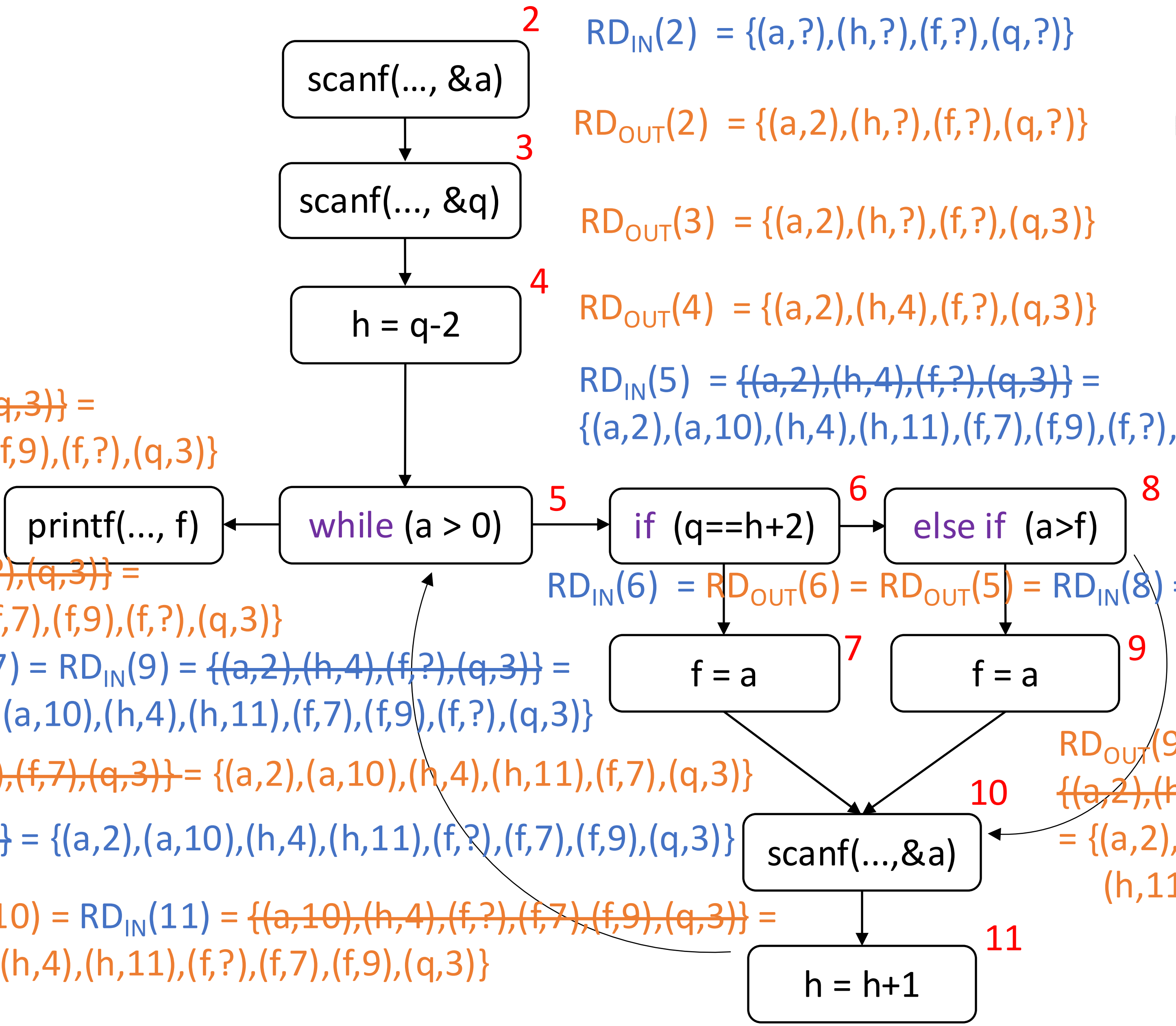
$RD_{IN}(10) = RD_{OUT}(7) \cup RD_{OUT}(8) \cup RD_{OUT}(9) = \{(a,2),(h,4),(f,?),(f,7),(f,9),(q,3)\}$

$RD_{OUT}(10) = RD_{IN}(11) = \{(a,10),(h,4),(f,?),(f,7),(f,9),(q,3)\}$

$RD_{OUT}(11) = \{(a,10),(h,11),(f,?),(f,7),(f,9),(q,3)\}$

Exercise 4

- Reaching definitions



RD$_{IN}$(2) = {(a,?),(h,?),(f,?),(q,?)}

RD$_{OUT}$(2) = {(a,2),(h,?),(f,?),(q,?)}

RD$_{OUT}$(3) = {(a,2),(h,?),(f,?),(q,3)}

RD$_{OUT}$(4) = {(a,2),(h,4),(f,?),(q,3)}

RD$_{IN}$(5) = ~~{(a,2),(h,4),(f,?),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}

RD$_{OUT}$(13) = ~~{(a,2),(h,4),(f,?),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}

RD$_{OUT}$(5) = ~~{(a,2),(h,4),(f,?),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}

RD$_{IN}$(6) = RD$_{OUT}$(6) = RD$_{OUT}$(5) = RD$_{IN}$(8) = RD$_{OUT}$(8)

RD$_{IN}$(7) = RD$_{IN}$(9) = ~~{(a,2),(h,4),(f,?),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}

RD$_{OUT}$(7) = ~~{(a,2),(h,4),(f,7),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,7),(q,3)}

RD$_{OUT}$(9) = ~~{(a,2),(h,4),(f,9),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,9),(q,3)}

RD$_{IN}$(10) = ~~{(a,2),(h,4),(f,?),(f,7),(f,9),(q,3)}~~ = {(a,2),(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)}

RD$_{OUT}$(10) = RD$_{IN}$(11) = ~~{(a,10),(h,4),(f,?),(f,7),(f,9),(q,3)}~~ = {(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)}

RD$_{OUT}$(11) = {(a,10),(h,11),(f,?),(f,7),(f,9),(q,3)}

Exercise 4

- Reaching definitions

$RD_{IN}(2) = \{(a,?),(h,?),(f,?),(q,?)\}$

$RD_{OUT}(2) = \{(a,2),(h,?),(f,?),(q,?)\} = RD_{IN}(3)$

$RD_{OUT}(3) = \{(a,2),(h,?),(f,?),(q,3)\} = RD_{IN}(4)$

$RD_{OUT}(4) = \{(a,2),(h,4),(f,?),(q,3)\}$

$RD_{IN}(5) = \{(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)\}$
$= RD_{OUT}(5) = RD_{IN}(6) = RD_{OUT}(6) = RD_{IN}(8) = RD_{OUT}(8)$
$= RD_{IN}(13) = RD_{OUT}(13)$

$RD_{IN}(7) = RD_{IN}(9) = \{(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)\}$

$RD_{OUT}(7) = \{(a,2),(a,10),(h,4),(h,11),(f,7),(q,3)\}$

$RD_{OUT}(9) = \{(a,2),(a,10),(h,4),(h,11),(f,9),(q,3)\}$

$RD_{IN}(10) = \{(a,2),(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)\}$

$RD_{OUT}(10) = \{(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)\} = RD_{IN}(11)$

$RD_{OUT}(11) = \{(a,10),(h,11),(f,?),(f,7),(f,9),(q,3)\}$

Exercise 4

$RD_{IN}(2)$ = {(a,?),(h,?),(f,?),(q,?)}

$RD_{OUT}(2)$ = {(a,2),(h,?),(f,?),(q,?)} = $RD_{IN}(3)$

$RD_{OUT}(3)$ = {(a,2),(h,?),(f,?),(q,3)} = $RD_{IN}(4)$

$RD_{OUT}(4)$ = {(a,2),(h,4),(f,?),(q,3)}

$RD_{IN}(5)$ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}
= $RD_{OUT}(5)$ = $RD_{IN}(6)$ = $RD_{OUT}(6)$ = $RD_{IN}(8)$ = $RD_{OUT}(8)$
= $RD_{IN}(13)$ = $RD_{OUT}(13)$

$RD_{IN}(7)$ = $RD_{IN}(9)$ = {(a,2),(a,10),(h,4),(h,11),(f,7),(f,9),(f,?),(q,3)}

$RD_{OUT}(7)$ = {(a,2),(a,10),(h,4),(h,11),(f,7),(q,3)}

$RD_{OUT}(9)$ = {(a,2),(a,10),(h,4),(h,11),(f,9),(q,3)}

$RD_{IN}(10)$ = {(a,2),(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)}

$RD_{OUT}(10)$ = {(a,10),(h,4),(h,11),(f,?),(f,7),(f,9),(q,3)} = $RD_{IN}(11)$

$RD_{OUT}(11)$ = {(a,10),(h,11),(f,?),(f,7),(f,9),(q,3)}

- UD chains
  - UD(q,4)={3}
  - UD(a,5)={2,10}
  - UD(f,13)={7,9,?}
  - UD(h,6)={4,11}
  - UD(q,6)={3}
  - UD(a,8)={2,10}
  - UD(f,8)={7,9,?}
  - UD(a,7)={2,10}
  - UD(a,9)={2,10}
  - UD(h,11)={4,11}

Exercise 4

- UD chains
  - UD(q,4)={3}
  - UD(a,5)={2,10}
  - UD(f,13)={7,9,?}
  - UD(h,6)={4,11}
  - UD(q,6)={3}
  - UD(a,8)={2,10}
  - UD(f,8)={7,9,?}
  - UD(a,7)={2,10}
  - UD(a,9)={2,10}
  - UD(h,11)={4,11}

- Def-use pairs
  - a: <2,5> <10,5> <2,8> <10,8> <2,7> <10,7> <2,9> <10,9>
  - h: <4,6> <11,6> <4,11> <11,11>
  - f: <7,13> <9,13> <?,13> <7,8> <9,8> <?,8> → potential use without definition
  - q: <3,4> <3,6>

Exercise 4

- Def-use pairs
  - f: <?,13> <?,8> → potential use without definition
- The pairs correspond to these execution paths:
  - <2 3 4 5 13>
  - <2 3 4 5 6 8 …>

```
0:   void main() {
1:     int a, h, f, q;
2:     scanf("%d", &a);
3:     scanf("%d", &q);
4:     h = q-2;
5:     while (a > 0) {
6:         if (q == h+2)
7:           f = a;
8:         else if (a > f)
9:           f = a;
10:        scanf("%d", &a);
11:        h = h+1;
12:    }
13:    printf("%d", f);
14: }
```

Exercise 4

- Symbolic execution <2 3 4 5 13>

```
0:   void main() {
1:     int a, h, f, q;
2:     scanf(“%d”, &a);
3:     scanf(“%d”, &q);
4:     h = q-2;
5:     while (a > 0) {
6:         if (q == h+2)
7:           f = a;
8:         else if (a > f)
9:           f = a;
10:        scanf(“%d”, &a);
11:        h = h+1;
12:    }
13:    printf(“%d”, f);
14: }
```

<0 1 2 3>

a	q	π
A	Q	T

<0 1 2 3 4>

a	q	h	π
A	Q	Q-2	T

<0 1 2 3 4 5 13>

a	q	h	π	SAT ✓
A	Q	Q-2	A≤0	

=> path is feasible, it’s an actual issue!

Exercise 4

- Symbolic execution <2 3 4 5 6 8 …>

```
0:   void main() {
1:     int a, h, f, q;
2:     scanf("%d", &a);
3:     scanf("%d", &q);
4:     h = q-2;
5:     while (a > 0) {
6:         if (q == h+2)
7:           f = a;
8:         else if (a > f)
9:           f = a;
10:        scanf("%d", &a);
11:        h = h+1;
12:    }
13:    printf("%d", f);
14: }
```

<0 1 2 3>

a	q	π
A	Q	T

<0 1 2 3 4>

a	q	h	π
A	Q	Q-2	T

<0 1 2 3 4 5>

a	q	h	π
A	Q	Q-2	A>0

<0 1 2 3 4 5 6 …>

a	q	h	π
A	Q	Q-2	A>0 Q≠Q-2+2

UNSAT ✗

=> path is unfeasible, it’s not an actual issue!