Run-time Environments

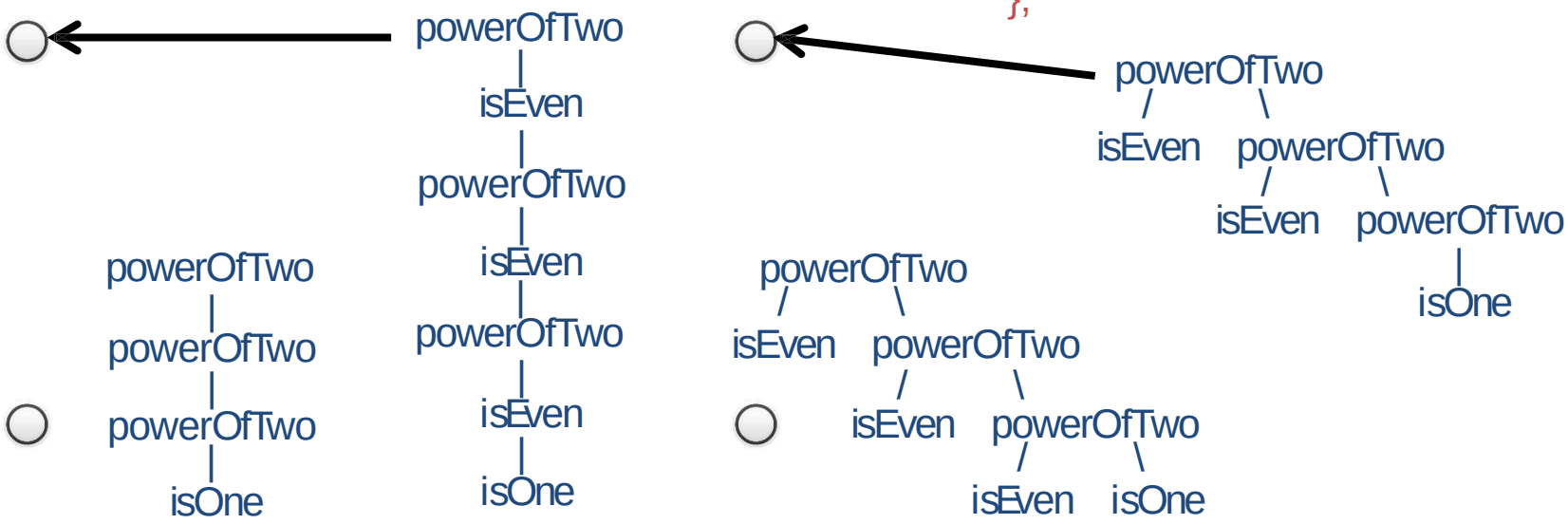
Lecture 10

Exercise

Question?

The powerOfTwo() function, shown to the right, returns true if its argument is a power of two, false otherwise. What is the activation tree for powerOfTwo(4)?

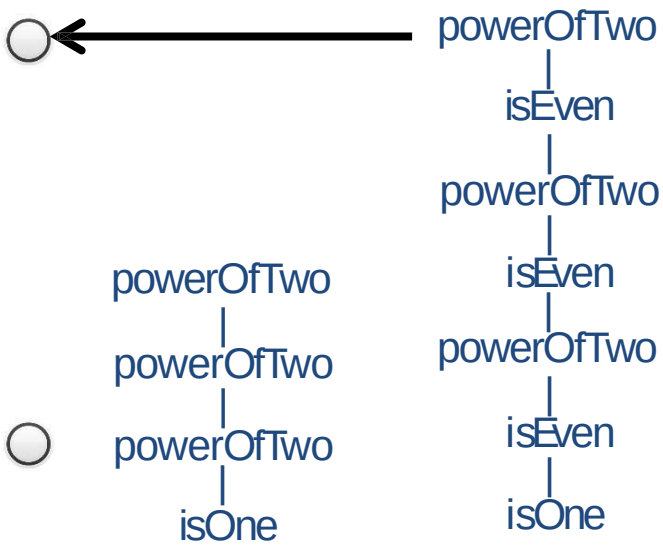
```
isEven(x:Int) : Bool { x % 2 == 0 };
isOne(x:Int) : Bool { x == 1 };
powerOfTwo(x:Int) : Bool {
   if isEven(x) then powerOfTwo(x / 2)
   else isOne(x)
};
```

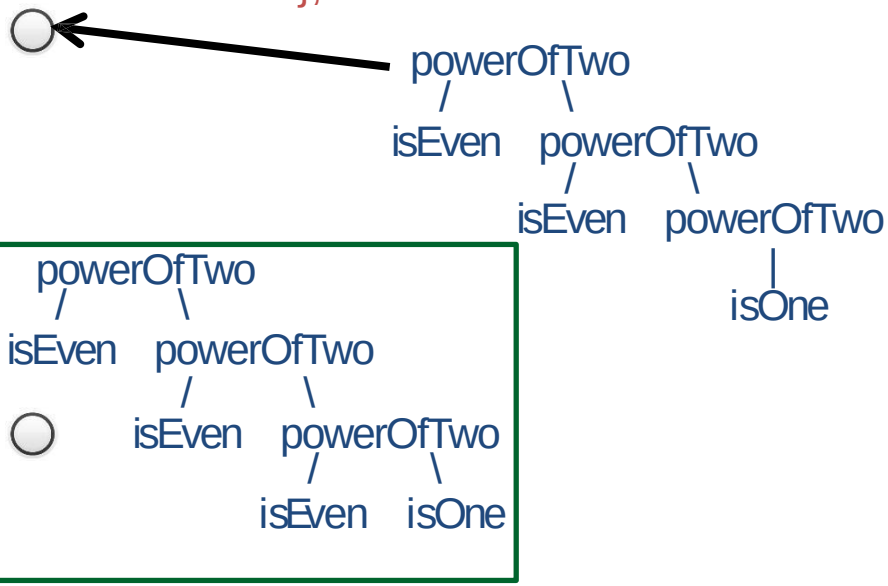


Answer!

The powerOfTwo() function, shown to the right, returns true if its argument is a power of two, false otherwise. What is the activation tree for powerOfTwo(4)?

```
isEven(x:Int) : Bool { x % 2 == 0 };
isOne(x:Int) : Bool { x == 1 };
powerOfTwo(x:Int) : Bool {
   if isEven(x) then powerOfTwo(x / 2)
   else isOne(x)
};
```





Question?

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```

Assume that in running the given program, the procedures are invoked in the order that follows: f (4), p (3), q (2), g(1), and f(1).

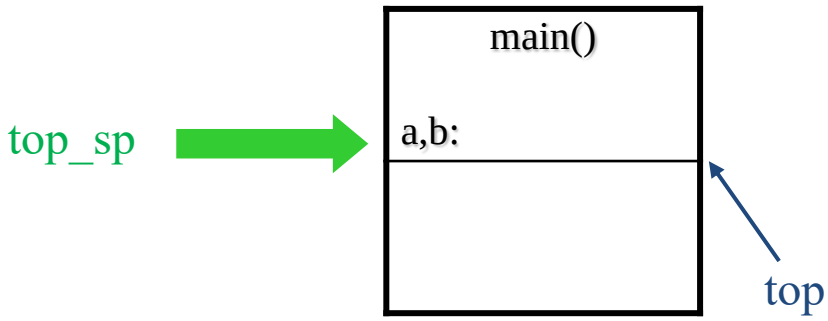
Draw the runtime stack at the end of these invocations. Assume that non-local variables are addressed using Access Links.

Assume there is any gap between access links and local data

What is the addresses of variables in assignment b := m + v in procedure q?

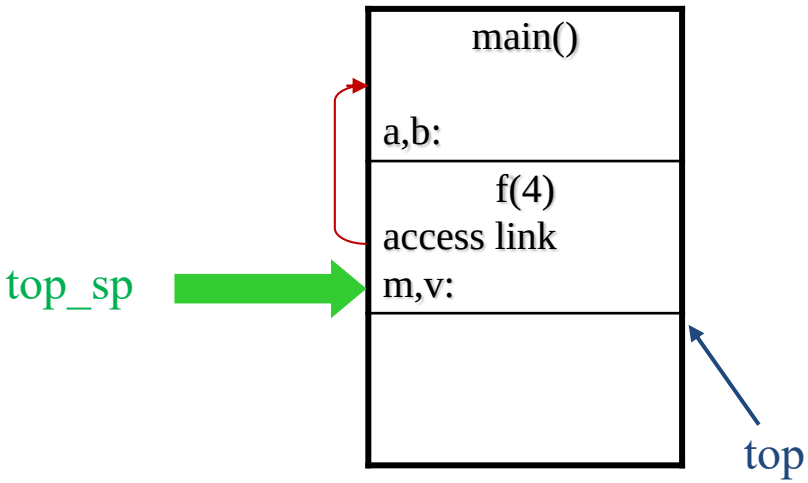
Answer!

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



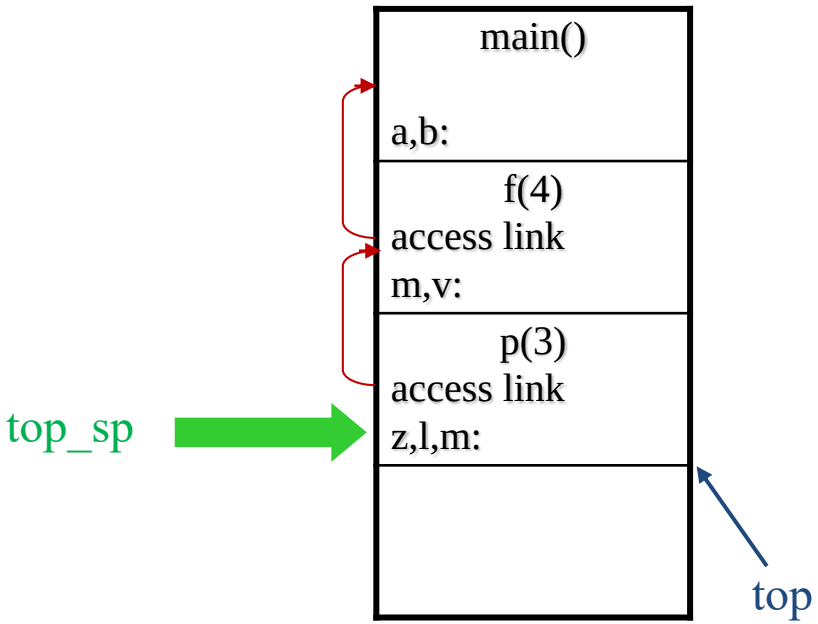
Answer! (cont.), when f(4) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



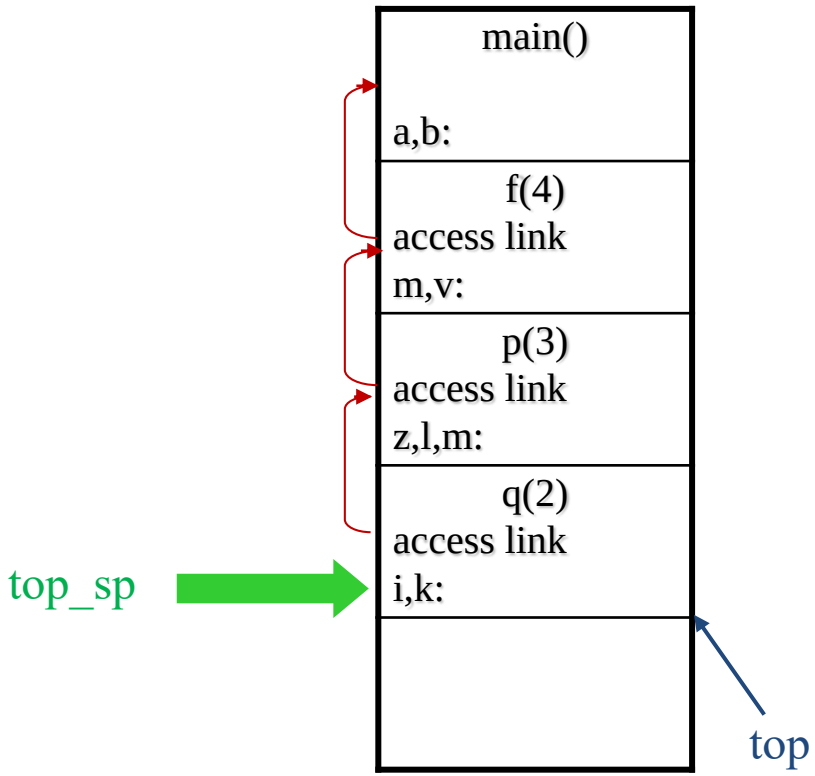
Answer! (cont.), when p(3) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



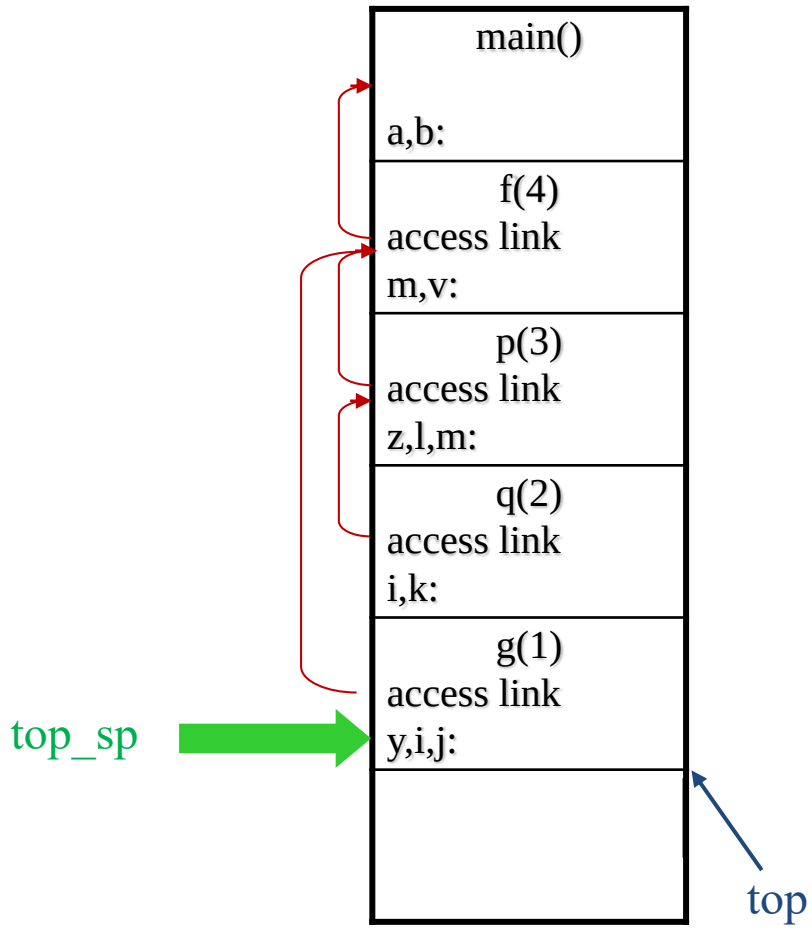
Answer! (cont.), when q(2) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



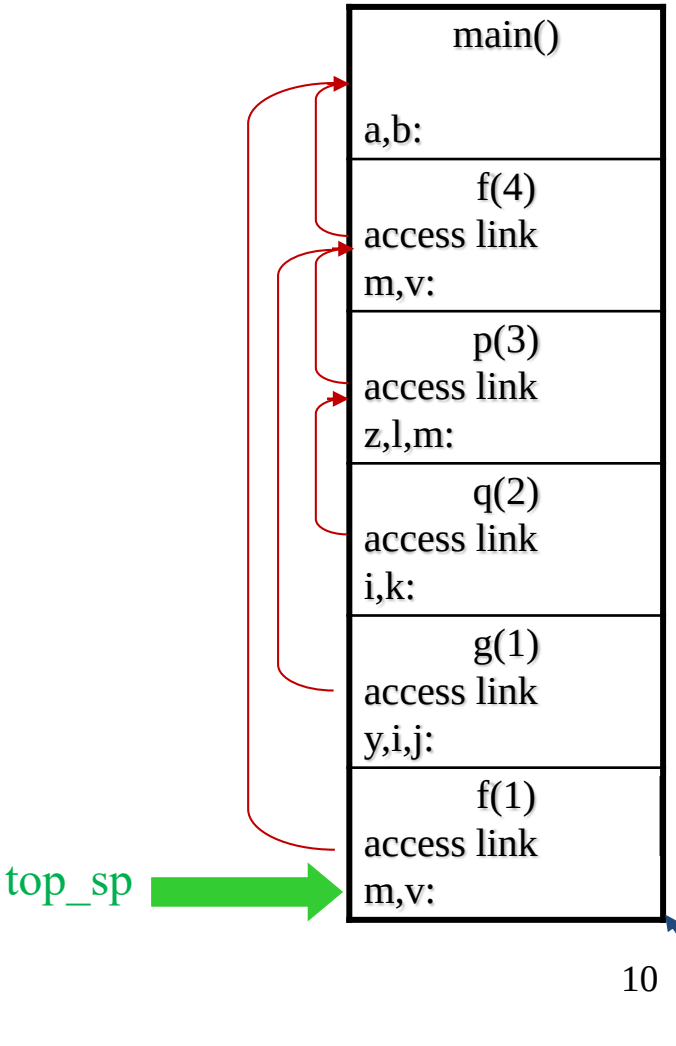
Answer! (cont.), when g(1) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



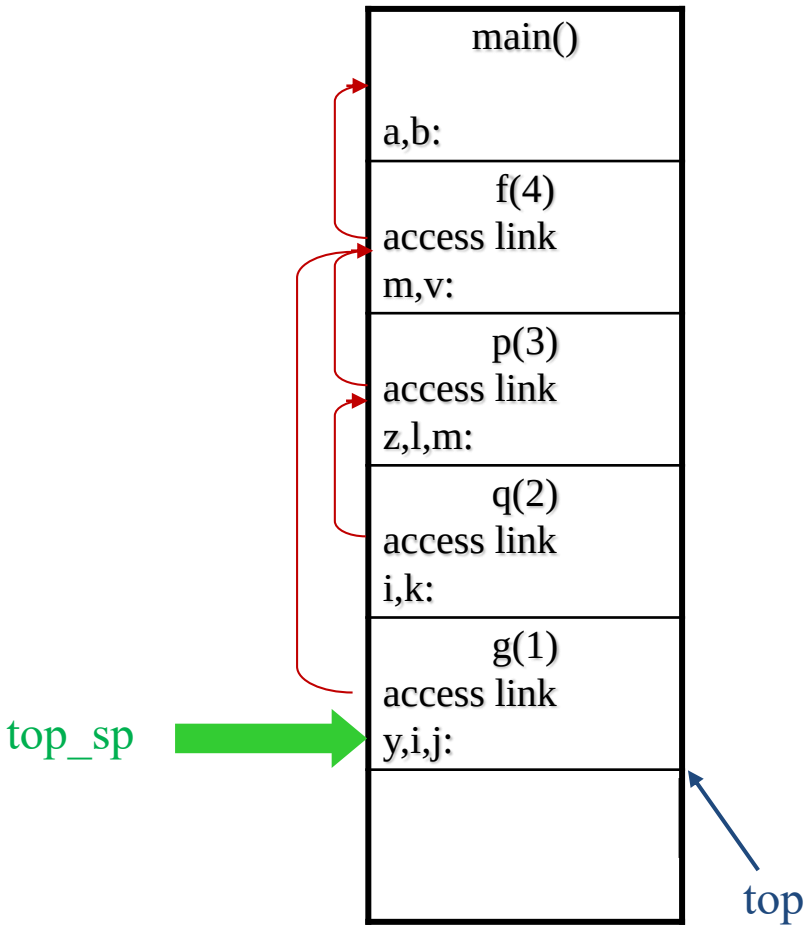
Answer! (cont.), when f(1) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



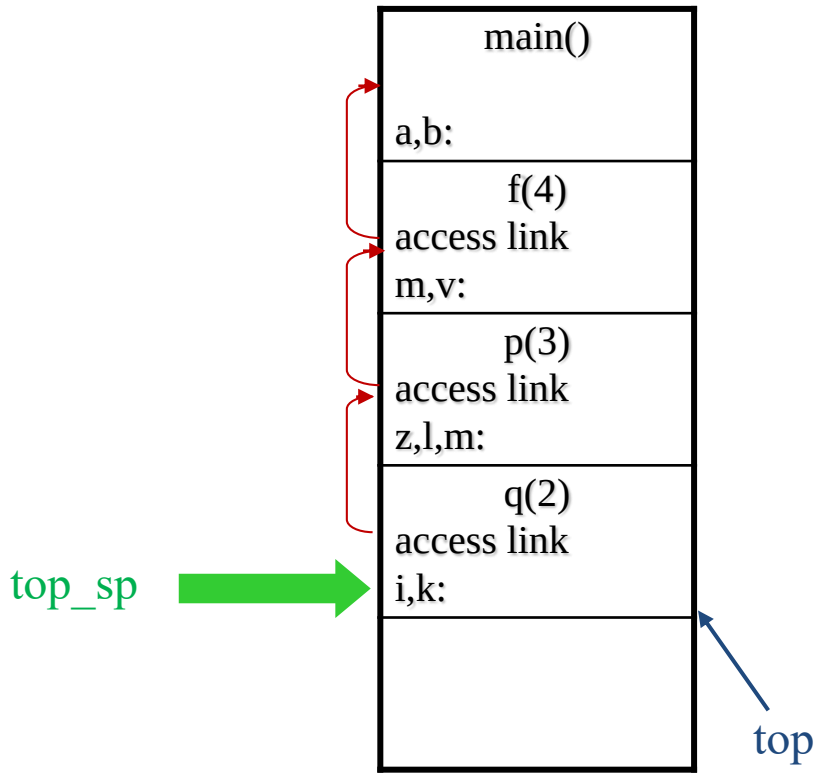
Answer! (cont.), when returned to g(1)

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



Answer! (cont.), when returned to q(2)

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



address m: @(top_sp - #1) + #1 + #2

address v: @(@(top_sp - #1)) + #1 + #1

address b: @(@(@(top_sp - #1))) + #1 + #1

Question?

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```

Assume that in running the given program, the procedures are invoked in the order that follows: f (4), p (3), q (2), g(1), and f(1).

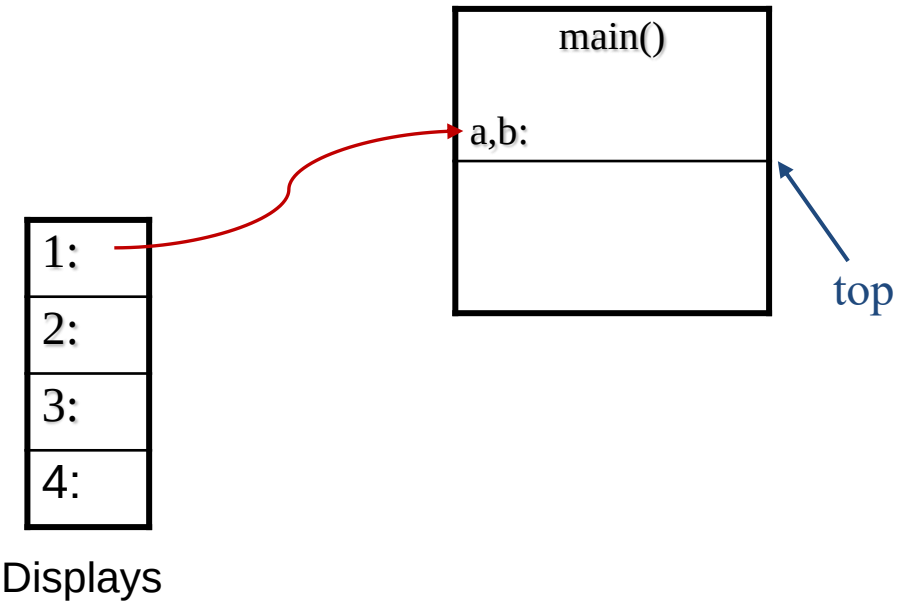
Draw the runtime stack at the end of these invocations. Assume that non-local variables are addressed using Displays.

Assume there is any gap between access links and local data

What is the addresses of variables in assignment b := m + v in procedure q?

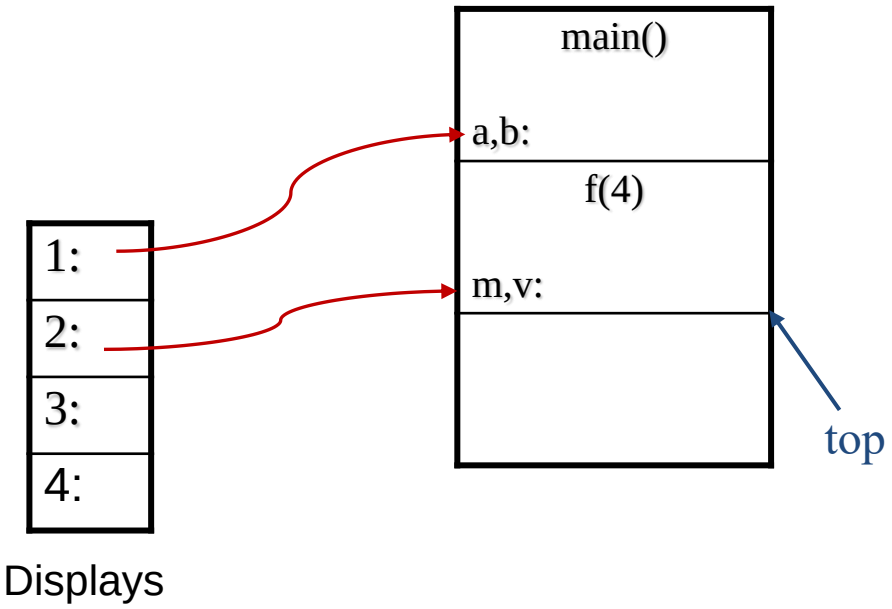
Answer!

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



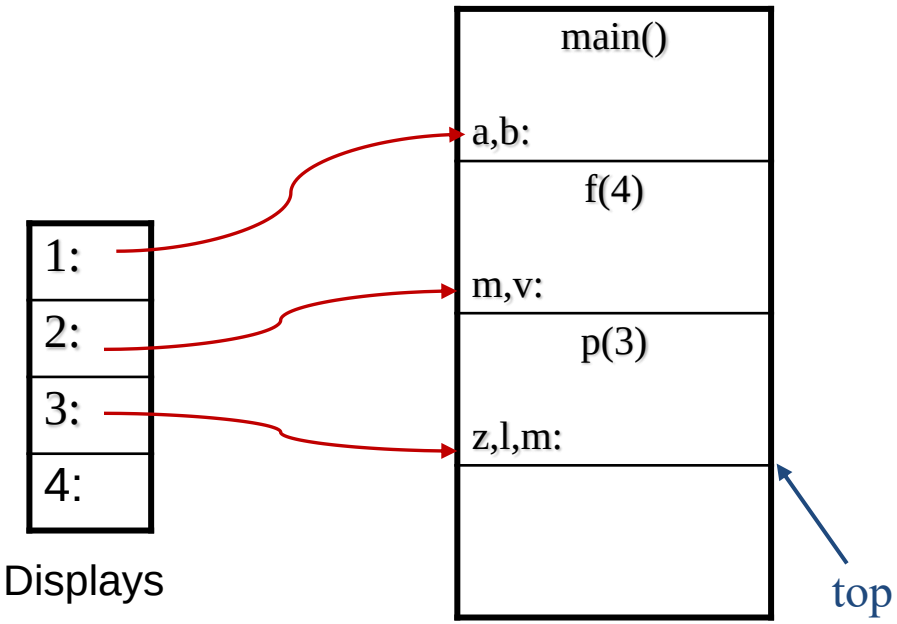
Answer! (cont.), when f(4) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



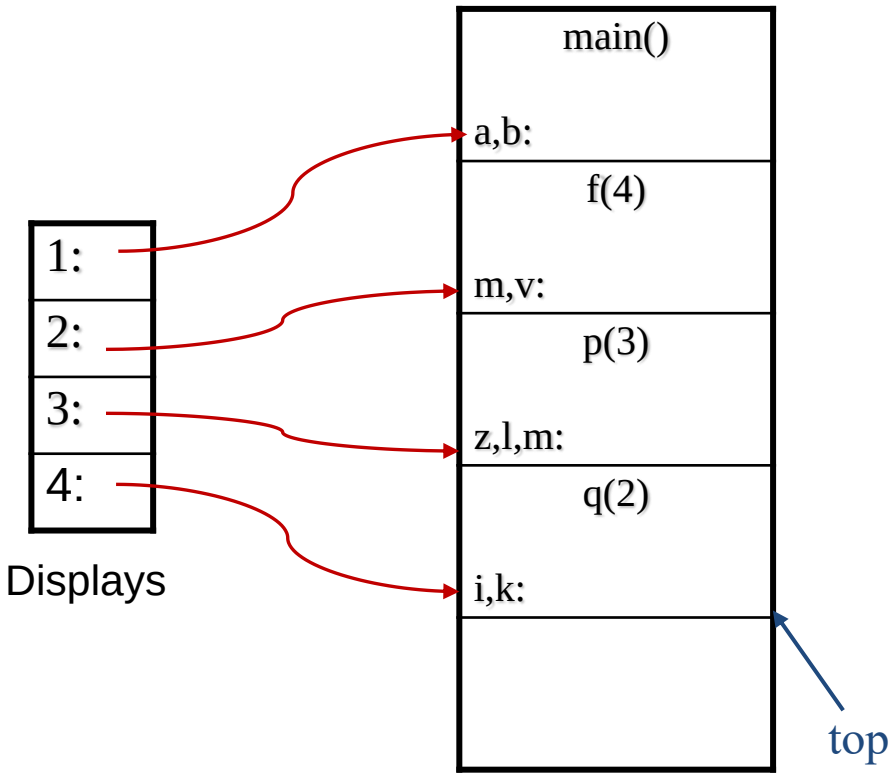
Answer! (cont.), when p(3) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



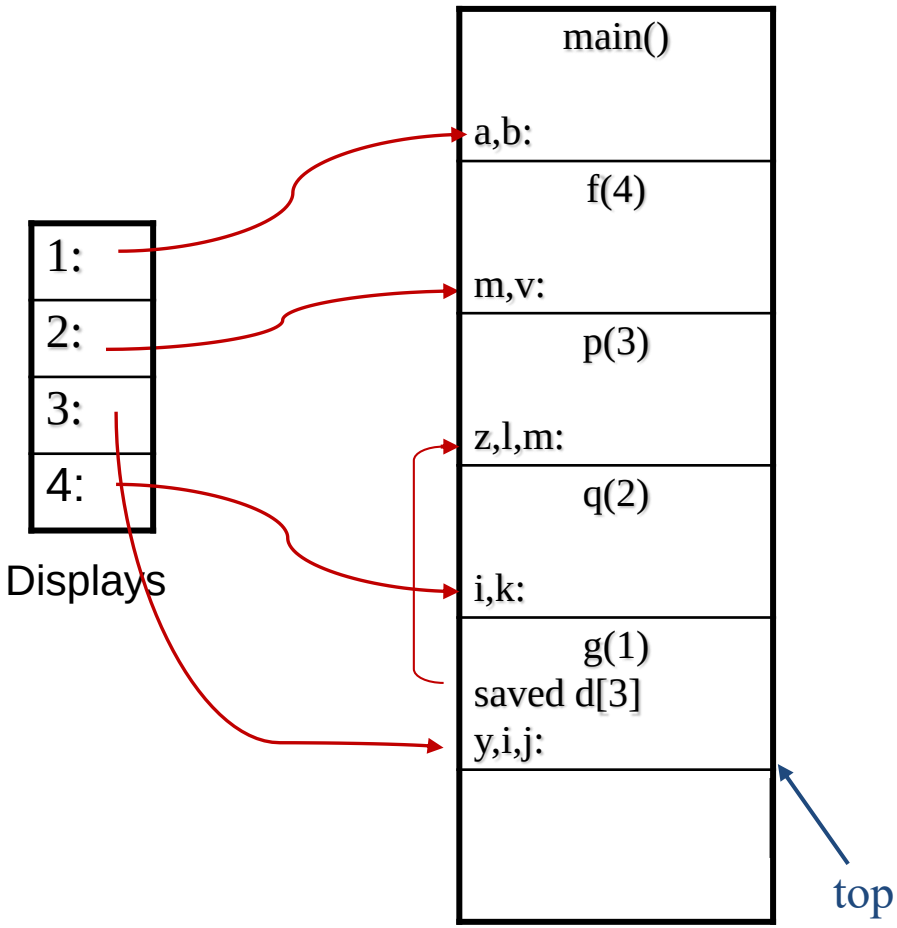
Answer! (cont.), when q(2) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



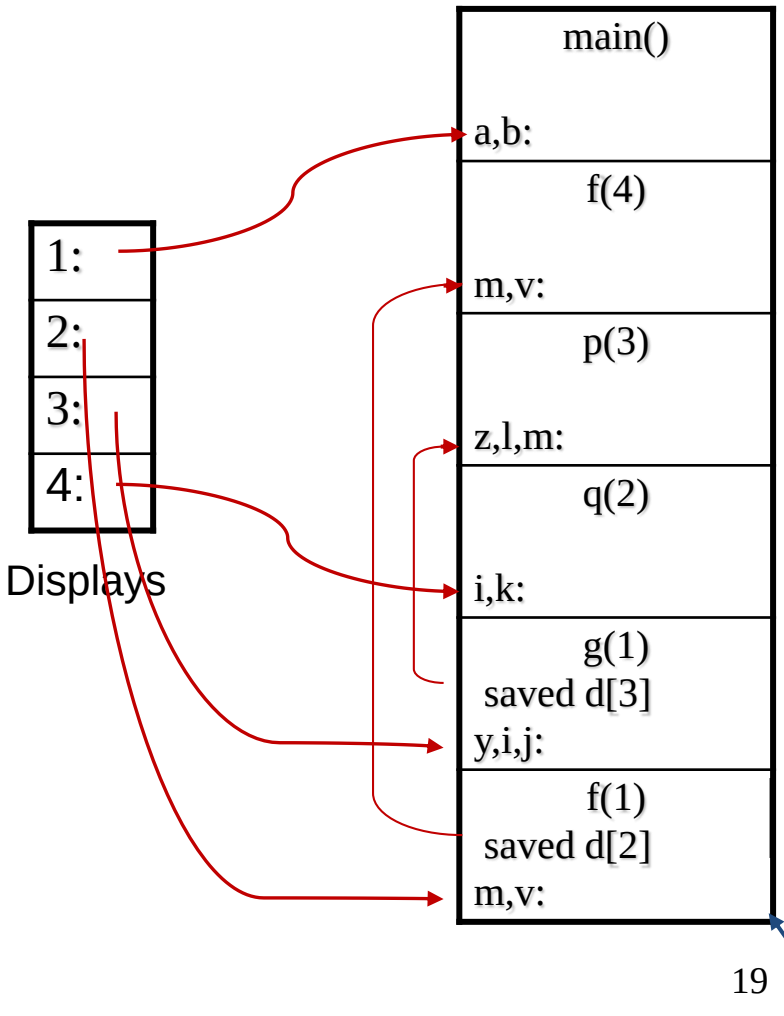
Answer! (cont.), when g(1) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



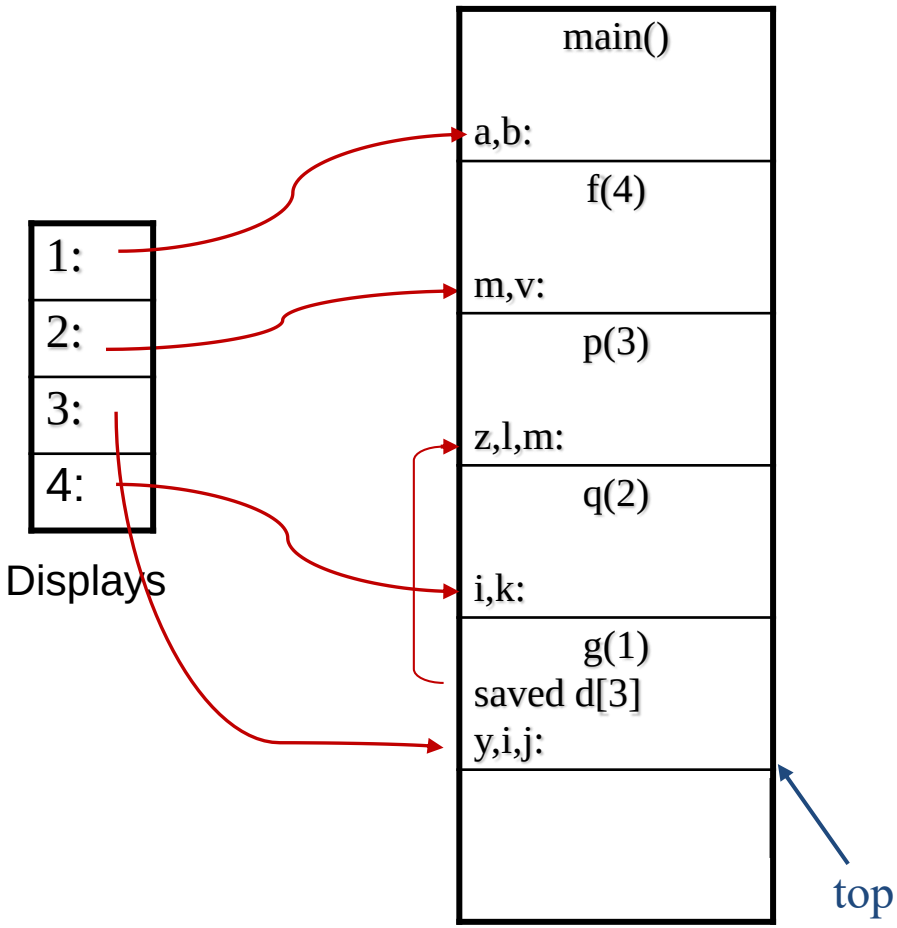
Answer! (cont.), when f(1) is called

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



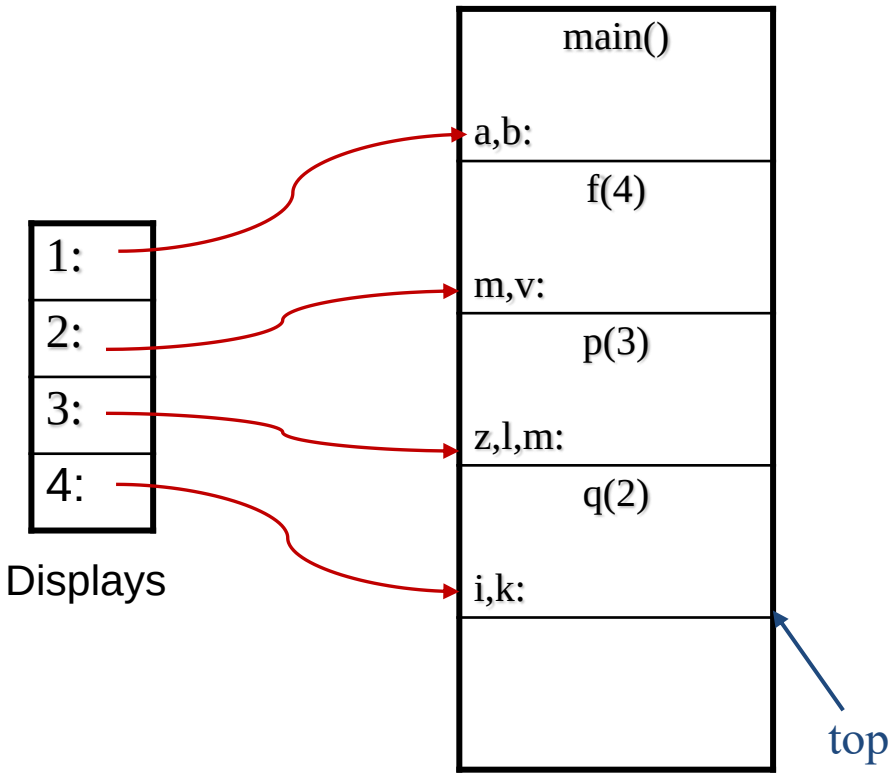
Answer! (cont.), when returned to g(1)

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



Answer! (cont.), when returned to q(2)

```
program main ()
    var a, b : int;
    procedure f (m : int);
        var v: int;
        procedure g (y : int);
            var i, j : int;
            f (y);
        end g;
        procedure p (z : int);
            var l, m : int;
            procedure q (i : int);
                var k : int;
                g (i - 1);
                b := m + v;
            end q;
            if (z > 4 ) f (z) else q (z - 1)
        end p;
        if  m > 1 then p (3)
    end f;
    f (4)
end main;
```



address m: D[3] + #2

address v: D[2] + #1

address b: D[1] + #1