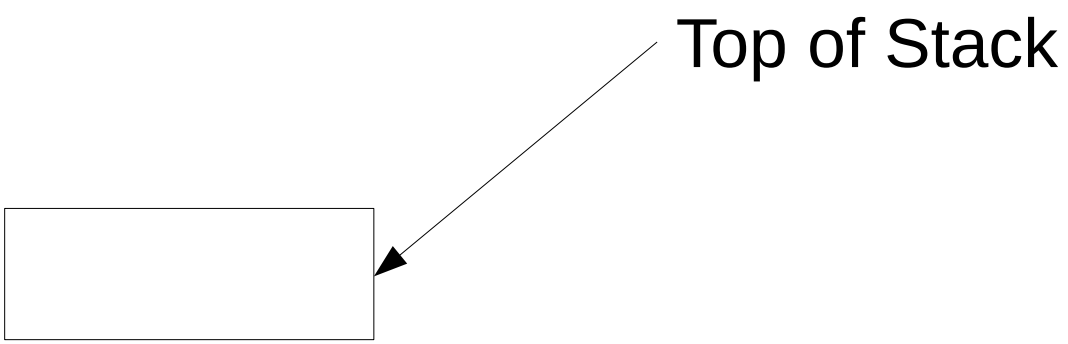
Scheme Heap

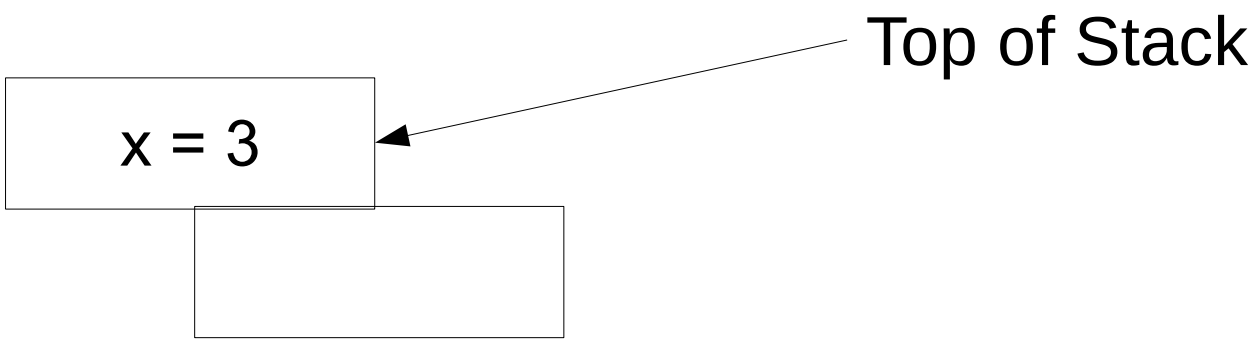
```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))
```



Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))
```

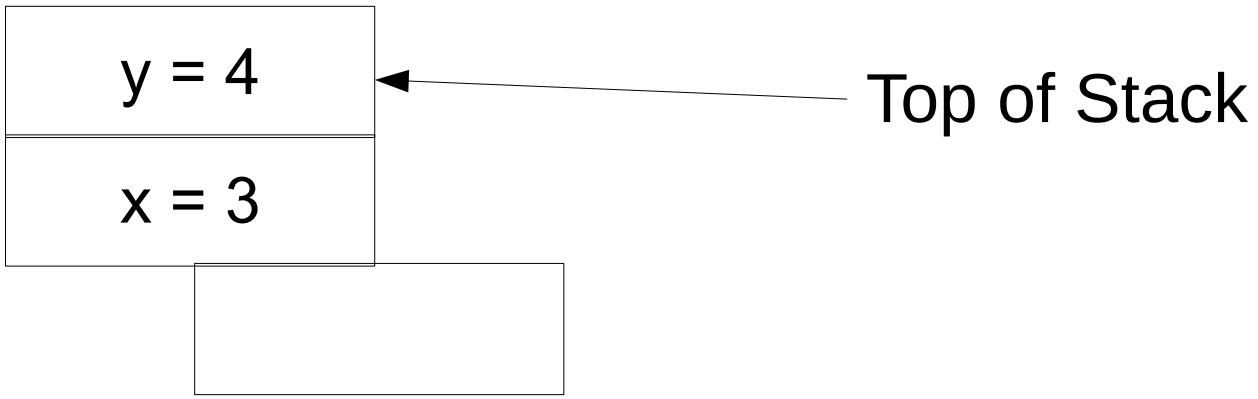


Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))
```



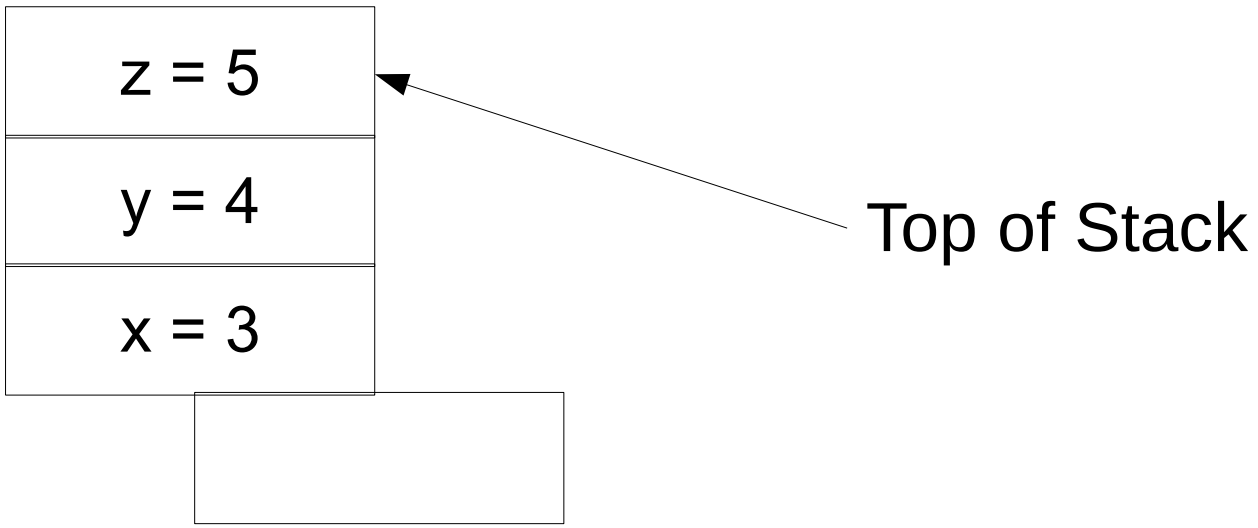
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))

(define f4
  (lambda ()
    (let ((z 5))
      'done)))
```



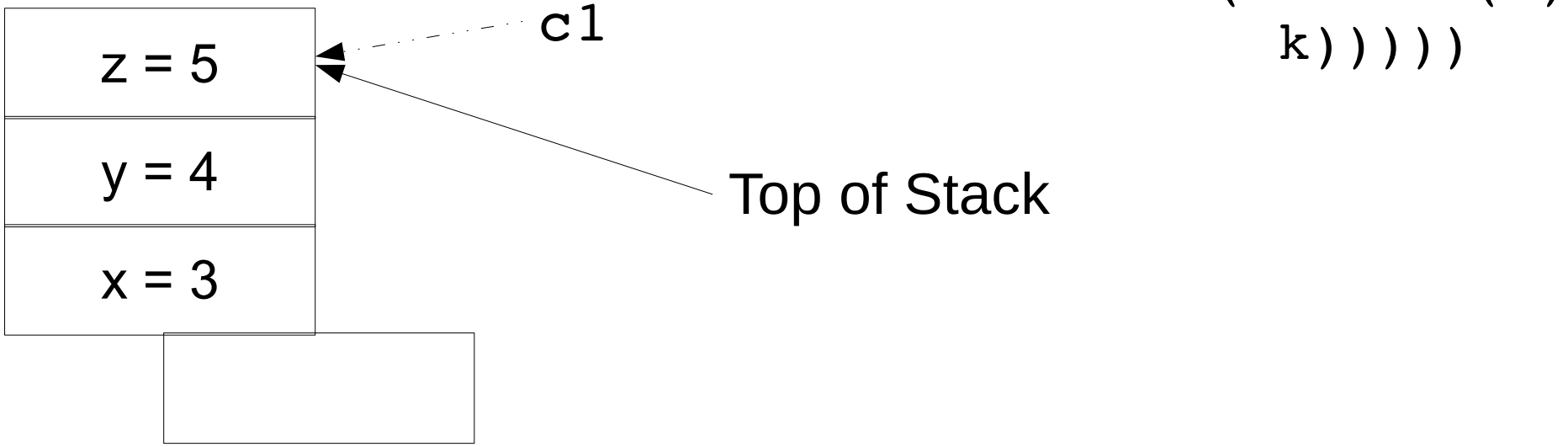
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



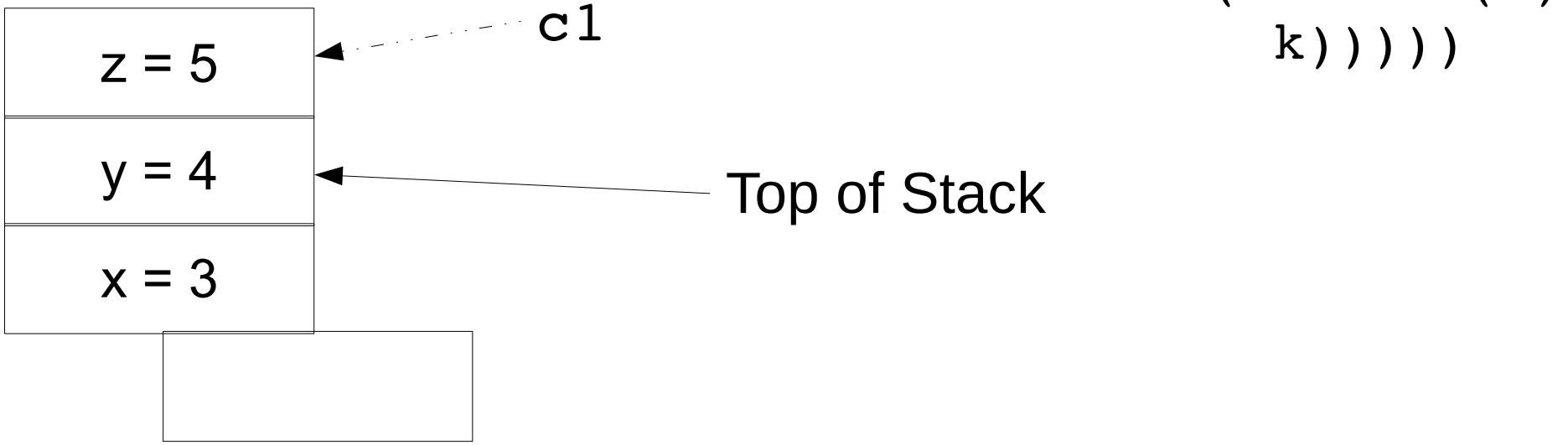
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      <cont>)))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



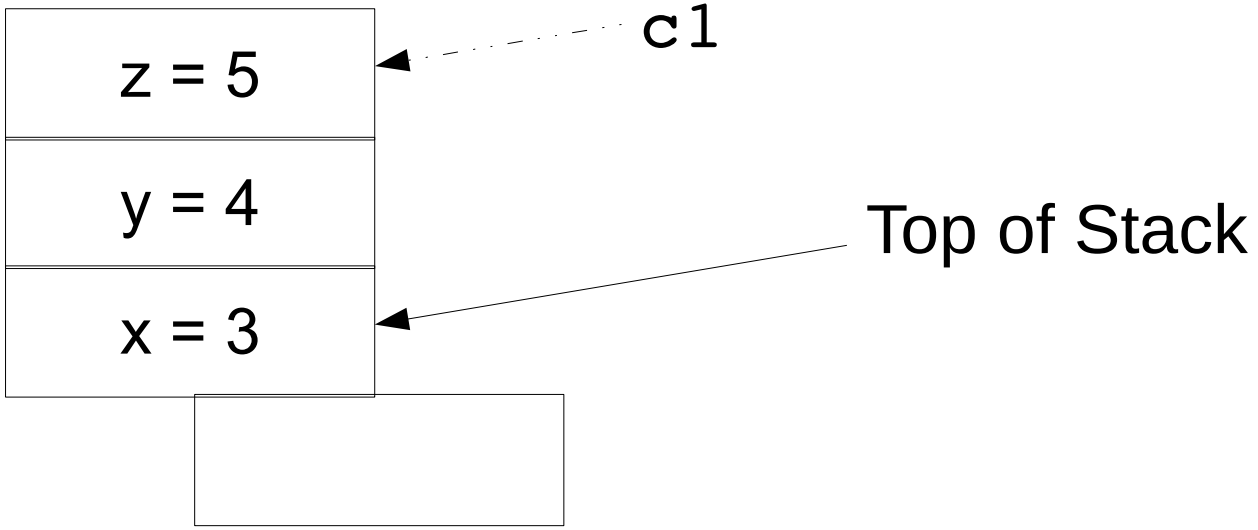
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 (f2))
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      <cont>)))

(define f3
  (lambda ()
    (let ((y 4))
      <cont>)))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



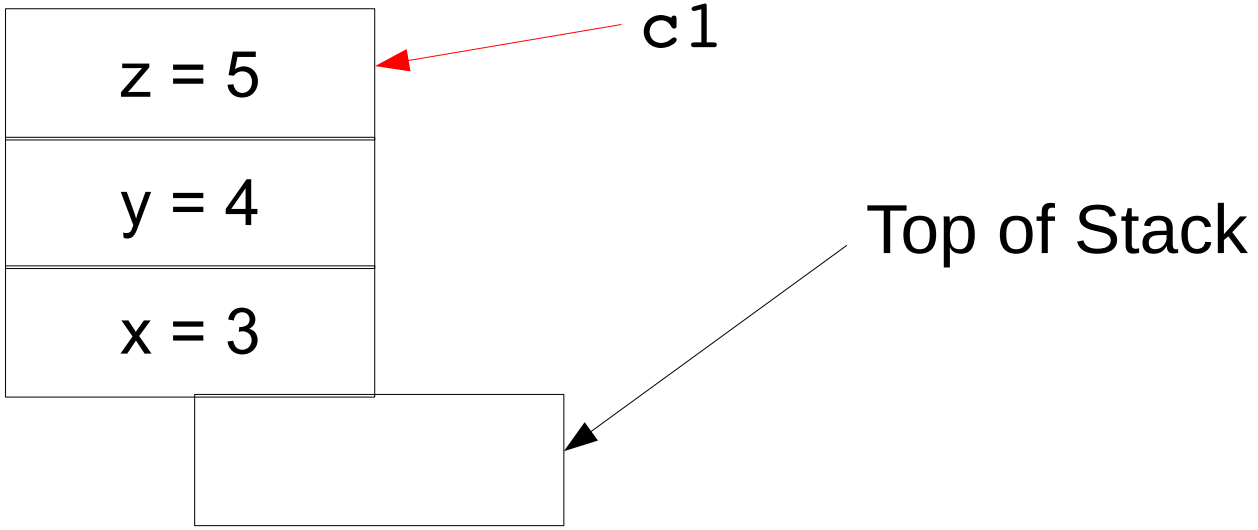
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      <cont>)))

(define f3
  (lambda ()
    (let ((y 4))
      <cont>)))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



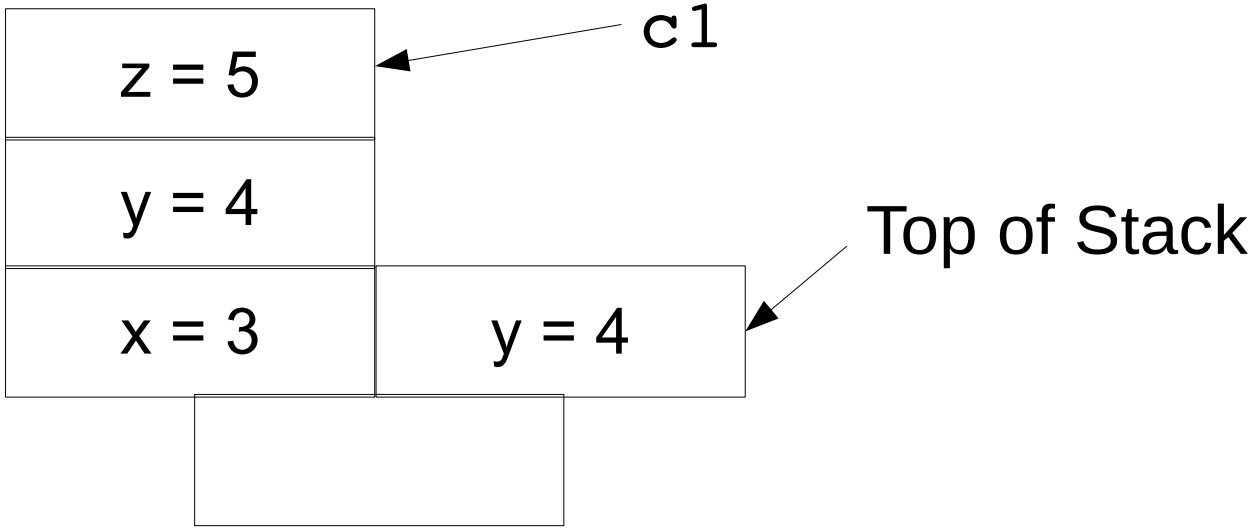
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



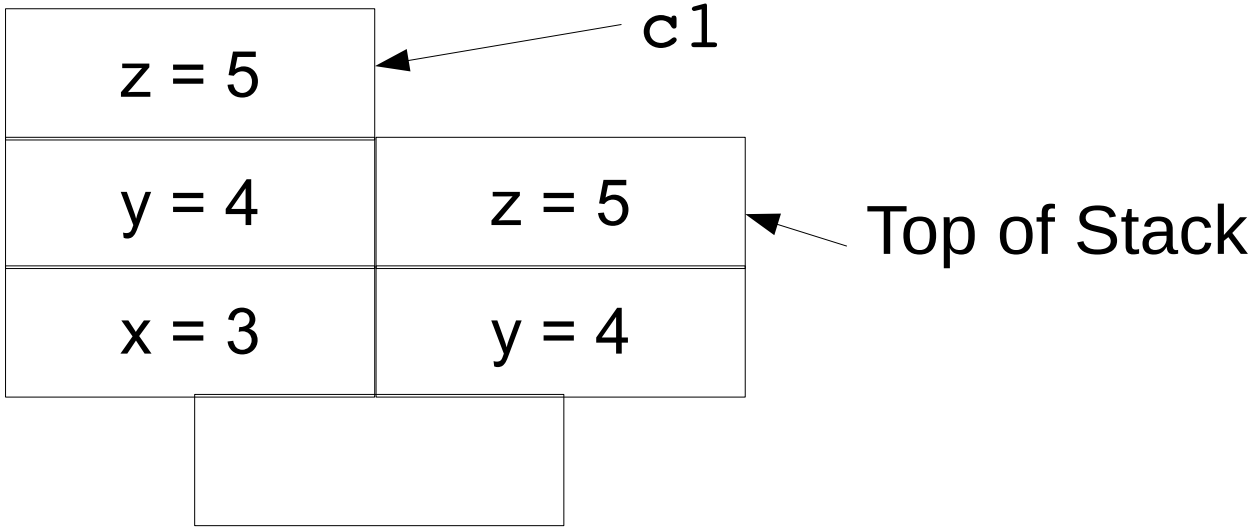
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



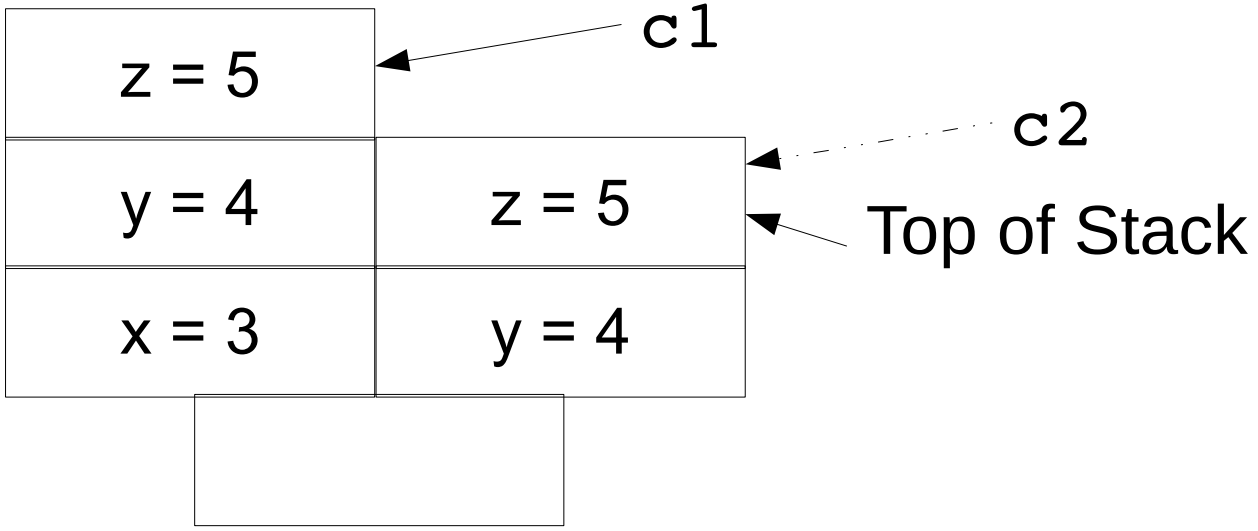
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      (f4))))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



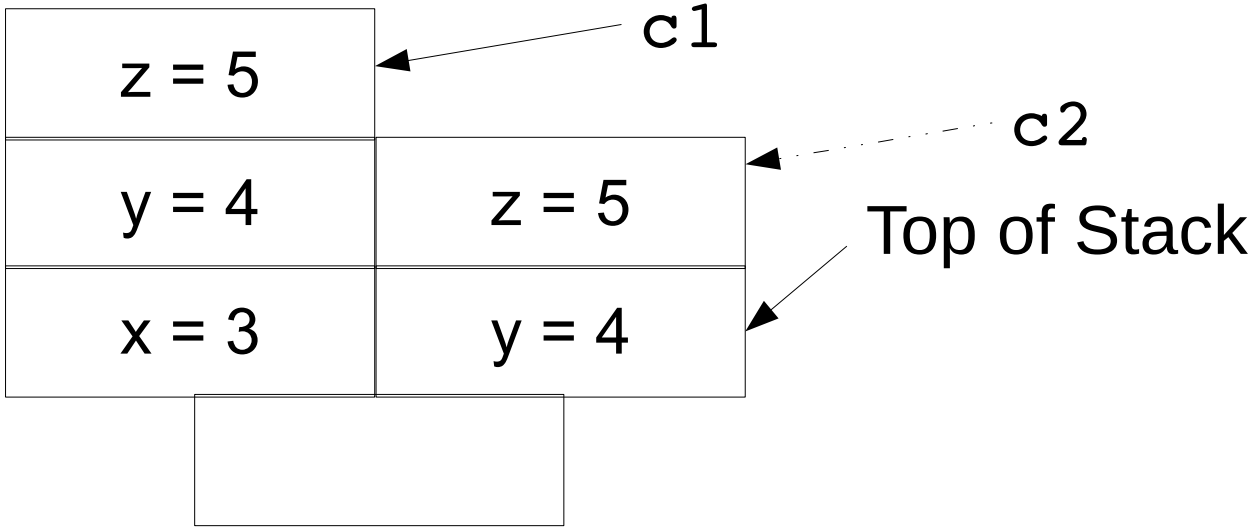
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 (f3)))
      (cons c1 c2))))

(define f2
  (lambda ()
    (let ((x 3))
      (f3))))

(define f3
  (lambda ()
    (let ((y 4))
      <cont>)))

(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



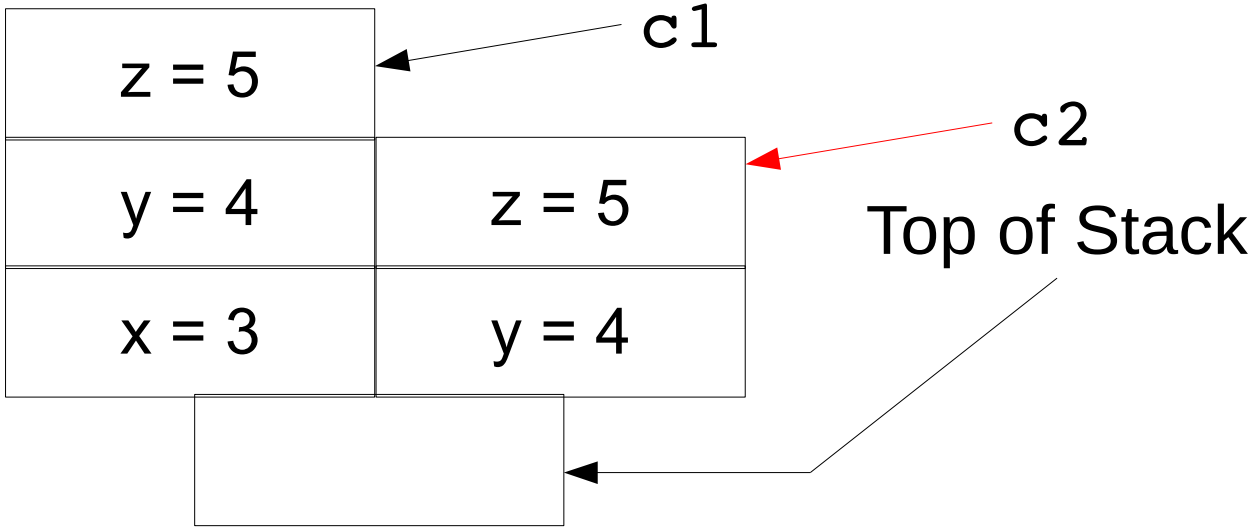
Scheme Heap

```
(define f1
  (lambda ()
    (let ((c1 <cont>)
          (c2 <cont>))
      (cons c1 c2))))
```

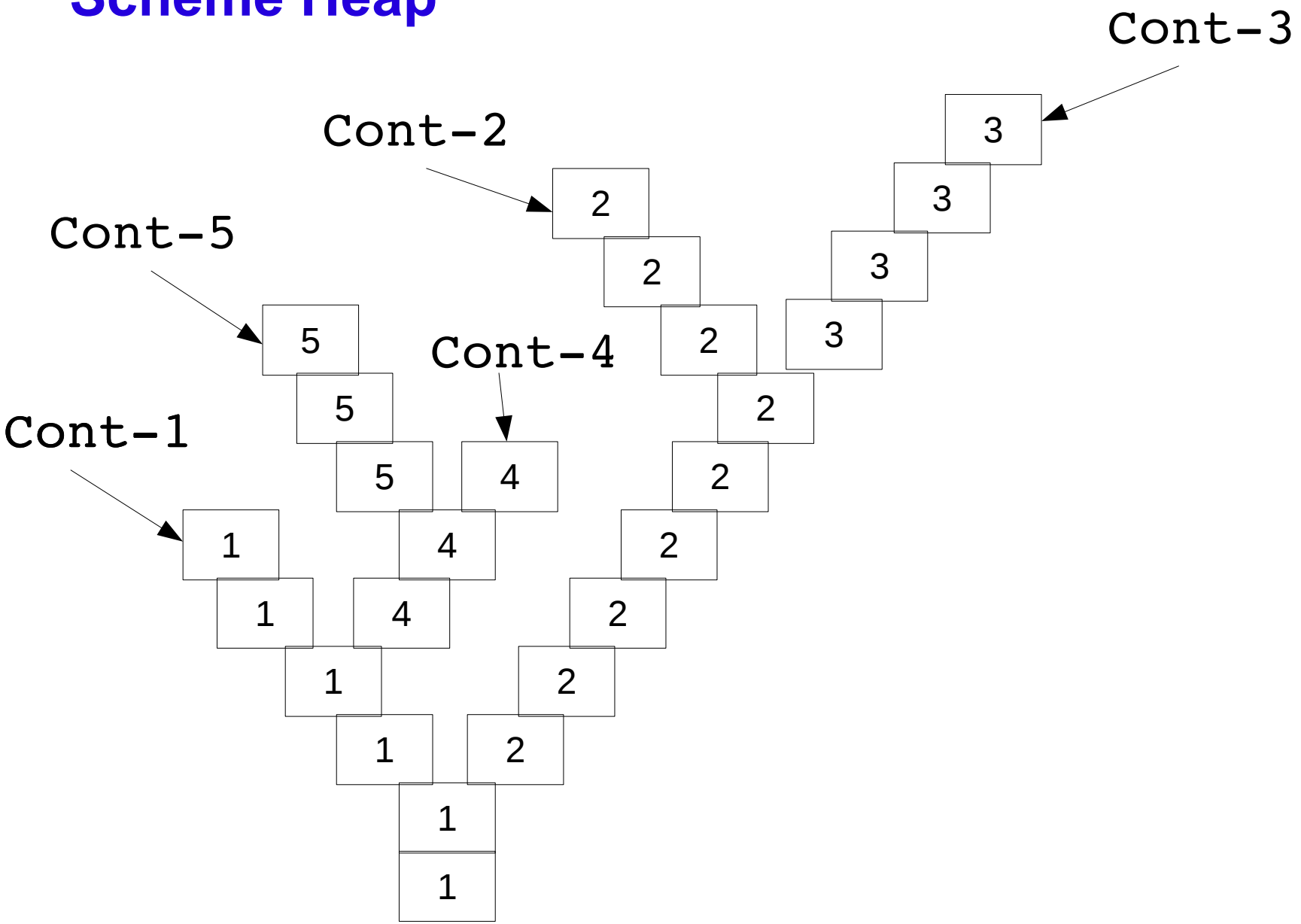
```
(define f2
  (lambda ()
    (let ((x 3))
      (f3))))
```

```
(define f3
  (lambda ()
    (let ((y 4))
      <cont>)))
```

```
(define f4
  (lambda ()
    (let ((z 5))
      (call/cc
        (lambda (k)
          k)))))
```



Scheme Heap


Cont-3
Cont-2
Cont-5
Cont-4
Cont-1
3
3
3
3
2
2
2
2
5
5
5
4
1
4
1
4
1
1
1
2
2
2
2
2
2
2
1
1
1