Respuestas: Gustavo Gutierrez Navarro

Mario Alejandro Castro Lerma, Tarea 1, Lenguajes de Programación

```
#lang stacker/smol/fun
(defvar x 2)
(deffun (addy y)
  (defvar x 1)
  (+ x y))
(addy 3)
x
```

Regresa 4 y 2

Es correcto, regresa 4 y 2 por que la suma que se realiza dentro de la función utiliza a la variable x local y no a la del bloque superior.

```
#lang stacker/smol/fun
(defvar x 2)
x
```

Regresa 2

Regresa 2, es trivial.

```
#lang stacker/smol/fun
(deffun (addy y)
  (defvar x 1)
  (+ x y))
(addy 4)
```

Regresa 5

Es una simple operación usando las variables, efectivamente da 5.

Si la variable es declarada en el bloque actual, usamos esa declaración.

```
#lang stacker/smol/fun
(defvar x 1)
(deffun (addy y)
  (+ x y))
(addy 4)
```

Regresa 5

Regresa 5 ya que al no encontrar la variable x dentro del bloque de la función addy lo busca en el bloque superior donde lo encuentra ligado con el valor de 1.

```
#lang stacker/smol/fun
(defvar y 9)
(deffun (addy)
  (+ x y))
(defvar x 1)
(addy)
```

Regresa 10

Similar al caso anterior , este programa logra regresar 10 debido a que la asignación de la variable x se da justo antes de llamar a la función addy la cual utilizara precisamente esta variable en su operación.

```
#lang stacker/smol/fun
(deffun (addy y)
  (+ x y))
(defvar x 1)
(addy 4)
```

Regresa 5

Si, regresa 5. Es exactamente igual que el caso anterior.

De lo contrario, buscamos en el bloque donde aparece el bloque actual, continuando recursivamente.

```
#lang stacker/smol/fun
(deffun (addy)
  (+ x y))
(addy)
```

Regresa error

Regresa error porque la función addy con 0 parámetros, ocupa dos variables en su operación de las cuales ninguna esta declarada, mas precisamente explota la intentar encontrar la variable x y no encontrarla.

```
#lang stacker/smol/fun
(deffun (addy y)
  (+ y z))
(addy 4)
```

Regresa error

Regresa error al momento de buscar la declaración de la variable z y no encontrarla.

```
#lang stacker/smol/fun
(defvar x 7)
(defvar y 9)
(deffun (search)
  x
  y
  z)
(search)
```

Regresa error

La función al momento de ejecutar la función search explota al intentar encontrar la variable z y fallar en el intento.

Si el bloque actual ya es el bloque primordial y aún no hemos encontrado una declaración correspondiente, la variable no está ligada.

```
#lang stacker/smol/fun
(defvar x 5)
(deffun (addy y)
  (defvar a 4)
  (+ x y))
(defvar b (addy (+ 1 2))
(defvar z (addy (+ 5 2))
z
```

Regresa 12

Regresa 12 por que el lugar donde se define a la variable z, se da de la llamada a la función addy con el parámetro 7 (resultado de la operación 5 + 2), en donde la función addy regresa el valor de x sumado a y, dichas variables se encuentran definidas en diferentes bloques, mientras que la variable y se encuentra dentro de la propia función con el valor de 7 que fue pasado como parámetro, el valor de x se puede encontrar en el bloqiue padrte de la función con el valor de 5, por lo tanto al devolver la suma devuelve efectivamente 12.

```
#lang stacker/smol/fun
(deffun (addy y)
  (defvar a x)
  (+ a y))
(defvar x (+ 15 20))
(defvar z (addy x))
z
```

Regresa 70

Al asignarle un valor a la variable z se llama la función addy con parámetro x, dicha x se definió previamente como 35 por lo tanto es el valor que se pasa al a función como y (dentro de la función). Durante la ejecución de addy se crea una nueva variable denominada como a que tendrá el valor de x (35) a la cual se le sumara el valor de y (35) y finalmente se devolverá su suma (35 + 35) finalmente devolviendo 70.

```
#lang stacker/smol/fun
(defvar x (+ 4 4))
(deffun (addy)
  (defvar y (+ 1 1))
  (+ x y))
(defvar z (addy))
z
```

Regresa 10

Se presenta el caso similar, dentro de la función tendremos a la variable y como 2 y a la variable x que se encuentra fuera de la función como 8 y al sumarlos devuelve 10.

1. Cada defvar evalua la expresión inmediatamente y vincula la varialbe al valor, incluso si la variable no es usada en el resto del programa.
2. Cada llamada a función evalúa los argumentos inmediatamente y vincula los valores a los parámetros formales, incluso cuando los parámetros formales no son usados en la función.

```
#lang stacker/smol/fun
(deffun (addy y)
  (+ x y))
(addy 10)
(defvar x 5)
```

Regresa Error

Resulta en un error por que se llama a la función addy la cual usa una variable x que no se encuentra declarada en su propio bloque local, por lo tanto al intentar buscarlo afuera se va encontrar con la declaración de la variable, sin embargo como todavía no se ha llegado a esa línea de código, esta variable tiene en su interior una bomba que al intentar leerla termina en una explosión del programa.

```
#lang stacker/smol/fun
(deffun (addy x)
  (+ x y)
  (defvar y 5))
(addy 5)
```

Regresa Error

Similar al caso anterior, únicamente que ahora la variable que contiene el explosivo se encuentra dentro de la misma función, por lo tanto al intentar ver su contenido provocara una explosión dentro del programa.

```
#lang stacker/smol/fun
(deffun (main)
  (defvar x y)
  (defvar y 5)
  x)
(main)
```

Regresa Error\

Cuando se llama a la función main, exactamente en la línea de defvar x y ocurre el error, esto debido a que se trata de usar la variable y, la cual como hemos visto en ejemplos anteriores, si todavía no se lee esa línea de código, la variable y existe como espacio de memoria, sin embargo no cuenta con un valor si no que esta contiene una bomba, lo cual termina en una explosión del programa.

1. Una secuencia de defvar y deffun vincula las variables a los valores de arriba hacia abajo y de izquierda a derecha.
2. Es un error evaluar una variable antes de vincularla a un valor.