Pregunta 5

a) i) Orden de evaluación normal

misteriosa "abc" (gen 1)

=>

foldr what const[] "abc" (gen 1)

=>

what "a" $ foldr what (const []) "bc" (1 : gen 1)

=>

("a", 1) $ foldr what (const[]) "bc" (gen 2)

=>

("a", 1) : what "b" $ foldr what (const[]) "c" (gen 2)

=>

("a", 1) : what "b" $ foldr what (const []) "c" (2 : gen 3)

=>

("a", 1) : ("b", 2) $ foldr what (const[]) "c" (gen 3)

=>

("a", 1) : ("b", 2) what "c" $ foldr what (const[]) "" (gen 3)

=>

("a", 1) : ("b", 2) what "c" $ foldr what (const []) "" (3 : gen 4)

=>

("a",1):("b",2):("c",3) $ foldr what (const []) "" (gen 4)

=>

("a",1):("b",2):("c",3) : (const []) "" (gen 4)

=>

("a",1) :("b",2) : ("c",3) : []

=>

("a",1) : ("b",2) :[("c",3)]

=>

("a",1) : [("b",2) , ("c",3)]

=>

[("a",1) , ("b",2) , ("c",3)]

ii) Orden aplicativo

misteriosa "abc" (gen 1)

=>

misteriosa "abc" (1 : gen 2)

=>

misteriosa "abc" (1 : 2 : gen 3)

Entra en evaluación infinita por que gen n no tiene limite superior.

b)
```
foldA f b Hoja = b
foldA f b Rama v RamaL RamaR =
    Rama (f value) (foldA f b RamaL)
                   (foldA f b RamaB)
```

c) i) Orden de evalcación normal

sospechosa arbolito (genA 1)

=>

foldA whatIF (const Hoja) arbolito (genA 1)

=>

foldA whatIF (const Hoja)
Rama 'a'
(Rama 'b' Hoja
(Rama 'c' Hoja Hoja)
)
Hoja) (genA 1)

=>

whatIF 'a' (foldA whatIF (const Hoja)
(Rama 'b'
Hoja
(Rama 'c' Hoja Hoja)
)
) (foldA whatIF (const Hoja) Hoja)
(genA 1)

=>

④

```
whatTF 'a' ( foldA whatTF (const Hoja)
                  (Rama 'b'
                        Hoja
                        (Rama 'c' Hoja Hoja)
                  )
           ) (foldA whatTF (const Hoja) Hoja)
           (Rama 1
                 (gen (1+1))
                 (gen (1*2))
           )
```

=>

```
Rama ('a',1) (( foldA whatTF (const Hoja)
                  (Rama 'b'
                        Hoja
                        (Rama 'c' Hoja Hoja)
                  )
             ) (genA (1+1)))
             ((foldA whatTF (const Hoja) Hoja)
               (genA (1*2)))
```

=>

```
Rama('a',1) (whatTF 'b' (foldA whatTF (const Hoja) Hoja)
             (foldA whatTF (const Hoja)
                   (Rama 'c' Hoja Hoja))
             (gen (1+1)))
              ((foldA whatTF (const Hoja) Hoja)
                (gen (1*2)))
```

=>

```
Rama('a',1) ( whatTF 'b'
              (foldA whatTF (const Hoja) Hoja)
              (foldA whatTF (const Hoja)
                            (Rama 'c' Hoja Hoja))
              (Rama 2 (gen (1+1)) (gen (2*2)))

              ((foldA whatTF (const Hoja) Hoja)
                 (gen (1*2))).
```

=>

```
Rama ('a',1) (Rama ('b',2)
    ((foldA whatTF (const Hoja) Hoja)
                   (gen (2+1)))
    (foldA whatTF (const Hoja)
                  (Rama 'c' Hoja Hoja)
                  (gen (2*2)))
  )((foldA whatTF (const Hoja) Hoja)
      (genA (1*2)))
```

=> Rama ('a',1)(

```
    Rama ('b',2)
        ((const Hoja) (genA (2+1)))
        ((foldA whatTF (const Hoja)
                       (Rama 'c' Hoja Hoja)
                       (gen (2*2)))
```

```
  )((foldA whatTF (const Hoja) Hoja)
      (gen (1*2)))
```

```
Rama ('a', 1)
    ( Rama ('b', 2)
           Hoja
           ( foldA whatTF (const Hoja)
                      (Rama 'c' Hoja Hoja )
                      (gen (2*2)))

) ( ( foldA whatTF (const Hoja) Hoja )
          (genA (1*2)))
```

=>

```
Rama ('a', 1) (
      Rama ('b', 2)
           Hoja
           (whatTF 'c' ( foldA whatTF (const Hoja)
                Hoja )
                ( fold whatTF (const Hoja)
                Hoja) ( gen (2*2)))
) ( ( foldA whatTF (const Hoja) Hoja )
          (gen (1*2)))
```

=>

```
Rama ('a', 1) (
      Rama ('b', 2)
           Hoja
           ( whatTF 'c' ( foldA whatTF (const Hoja)
                Hoja)
                ( foldA whatTF (const Hoja)
                (Hoja) Rama 4
                      (gen (4+1)) (gen (4*2)))

)(( foldA whatTF const Hoja) Hoja) ( gen (1*2)))
```

```
Rama ('a', 1) (
        Rama ('b', 2)
            Hoga
            ( Rama ('c', 4)
                (( foldA whatTF (const Hoga) Hoga)
                    (gen (4+1))
                (( foldA whatTF (const Hoga) Hoga)
                    gen ( 4*2))
            )
    ) (( foldA whatTF (const Hoga) Hoga)
        (gen ( 1*2)))
```

=>

```
Rama ( 'a', 1) (
        Rama ('b', 2)
            Hoga
            ( Rama ('c', 4)
                (( const Hoga) (gen (4+1)))
                (( foldA whatTF (const Hoga) Hoga)
                    (gen ( 4*2)))
            )
    ) (( foldA whatTF (const Hoga) Hoga)
        (genA (1*2)))
```

=>

```
Rama ('a', 1)(
        Rama('b',2)
           Hoja
           (Rama ('c', 4)
              Hoja
              (( foldA  whatTF (const Hoja) Hoja)
                        (gen (4*2)))
           )
    ) (( foldA  whatTF (const Hoja) Hoja)
               (gen (1*2)))
```

=>

```
Rama ( 'a', 1)(
        Rama ('b',2)(
           Hoja
           ( Rama ('c', 4)
              Hoja
              (const  Hoja)(gen (4*2))
           )
    )(( foldA  whatTF (const Hoja)  Hoja)
               (gen (1*2)))
```

=>

```
Rama  ('a', 1)(
        Rama ('b',2)( Hoja
           Rama ('c',4)  Hoja  Hoja
        )
    )
    (( foldA  whatTF  (const  Hoja)  Hoja)
              (gen  (1*2)))
```

=>

```
Rama ('a',1)(
        Rama ('b',2)
            Hoja
            Rama ('c',4)
                Hoja
                Hoja
) ((const Hoja) (gen (1*2))
```

=>

```
Rama ('a',1)
    Rama ('b',2)
        Hoja
        Rama ('c',4)
            Hoja
            Hoja
    Hoja
```

ii)

sospechosa arbolito (genA 1)

=>

sospechosa arbolito Rama 1
(genA (2))
(genA (2))

=>

sospechosa arbolito Rama 1
(Rama 2
(gen (3))
(gen (4))
(genA 2)

Entra en recursión infinita ya que genA n no tiene condición de parada.