# Pregunta 3a

•••

¡Muéstrales de qué está hecho un Saiyan!

## Codigo Con Sustitucion de Constantes

```
class Abra {
  int a = 2, b = 1
  fun cus(int x): int {
    a = b + x
    return pide(a)
  }
  fun pide(int y): int {
    return a - y * b
  }
  class Cadabra extends Abra {
    Abra zo = new PataDeCabra()
    fun pide(int y): int {
       return zo.cus(a + b) - y
    }
}
```

```
class PataDeCabra extends Cadabra {
  int b = 7, c = 6
  fun cus(int x): int {
    a = x - 3
    c = a + b * c
    return pide(a * b + x)
  }
  fun pide(int y): int {
    return c - y * a
}
```

Se consideran las variables a,b, c y zo como Variables de instancia dado que no son statics (similar a java)

### Asociación Estática de Métodos

Abra ho = new Cadabra()

Abra po = new PataDeCabra()

Cadabra cir = new PataDeCabra()

print(ho.cus(1) + po.cus(1) + cir.cus(1))

Inicialmente se crean dos variables de tipo Abra y una de tipo Cadabra

Por emplearse asociación estática de métodos se tiene que ho y po tendrán los métodos de Abra. Y dado que Cadabra extiende a Abra y sobre-escribe la función pide entonces cir tiene dos métodos: cus (de Abra) y pide (de Cadabra).

Como el espacio para visualizar la ejecución es limitado se mostrará únicamente la línea print y se mostrará el método que se esté ejecutando

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
   }
   fun pide(int y): int {
3      return a - y * b
   }
}
```

ho		po		ci	cir	
cus	Abra	cus	Abra	cus	Abra	
pide	Abra	pide	Abra	pide	Cad abra	

cus@Abra	х	1
	b	1
	а	2

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
   }
   fun pide(int y): int {
3      return a - y * b
   }
}
```

ho		po		cir	
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

cus@Abra	х	1
	b	1
	а	<u>ጂ</u> 2

```
fun cus(int x): int {
          a = b + x
           return pide(a)
       fun pide(int y): int {
          return a - y * b
   ho
                             cir
                 po
cus
      Abra
             cus
                   Abra
                           cus
                                 Abra
pide
      Abra
             pide
                   Abra
                           pide
                                 Cad
                                 abra
```

class Abra {

int a = 2, b = 1

	pide@Abr a	у	2
		b	1
		а	2
	cus@Abra	х	1
		b	1
		а	2 2

```
class Abra {
      int a = 2, b = 1
       fun cus(int x): int {
          a = b + x
          return pide(a)
      fun pide(int y): int {
                                Retorna
          return a - y * b 2-2*1 = 0
   ho
                            cir
                po
cus
     Abra
             cus
                   Abra
                          cus
                               Abra
pide
      Abra
             pide
                  Abra
                          pide
                                Cad
```

abra

	pide@Abr a	у	2
		b	1
		а	2
	cus@Abra	х	1
		b	1
		а	2 2

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
      }
   fun pide(int y): int {
3      return a - y * b
      }
}
```

po

Abra

Abra

cus

pide

ho

Abra

Abra

cus

cir					
cus	Abra				
pide	Cad abra				

cus@Abra	х	1
	b	1
	а	<u>ጀ</u> 2

```
print(ho.cus(1) + po.cus(1) + cir.cus(1))
                      ho.cus(1) = 0
fun cus(int x): int {
fun pide(int y): int {
```

ho		p	0	ciı	r
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

class Abra {

int a = 2, b = 1

a = b + x

return pide(a)

return a - y \* b

```
class Abra {
    int a = 2, b = 1
0    fun cus(int x): int {
1        a = b + x
2        return pide(a)
    }
    fun pide(int y): int {
3        return a - y * b
    }
}
```

po

cus

pide

ho

Abra

Abra

cus

)	ciı	-
Abra	cus	Abra
Abra	pide	Cad abra

```
class Abra {
    int a = 2, b = 1
0    fun cus(int x): int {
1         a = b + x
2         return pide(a)
     }
    fun pide(int y): int {
3         return a - y * b
     }
}
```

)	р	0	ci	r
Abra	cus	Abra	cus	Abra
Abra	pide	Abra	pide	Cac

cus

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
   }
   fun pide(int y): int {
3      return a - y * b
   }
}
```

ho		po		cir	
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

cus@Abra	х	1
	b	1
	а	2

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
   }
   fun pide(int y): int {
3      return a - y * b
   }
}
```

h	0	p	O		cir
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	e Cad abra
				<u> </u>	

	х	1
cus@Abra	b	1
	а	2⁄2

```
int a = 2, b = 1
       fun cus(int x): int {
          a = b + x
           return pide(a)
       fun pide(int y): int {
          return a - y * b
   ho
                             cir
                 po
                   Abra
cus
      Abra
             cus
                           cus
                                 Abra
                           pide
pide
      Abra
             pide
                   Abra
                                 Cad
                                 abra
```

class Abra {

	у	2
pide@Abr a	b	1
	а	2
	х	1
cus@Abra	b	1
	а	2 2

```
class Abra {
      int a = 2, b = 1
       fun cus(int x): int {
          a = b + x
          return pide(a)
      fun pide(int y): int {
                                Retorna
          return a - y * b 2-2*1 = 0
   ho
                            cir
                po
cus
     Abra
            cus
                  Abra
                         cus
                               Abra
pide
     Abra
            pide
                  Abra
                         pide
                               Cad
```

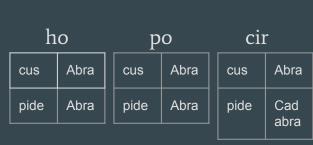
abra

		у	2
	pide@Abr a	b	1
		а	2
	cus@Abra	х	1
		b	1
		а	2 2

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
      }
   fun pide(int y): int {
3      return a - y * b
      }
}
```

h	O	p	O	ciı	(
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

	х	1
cus@Abra	b	1
	а	፮ 2



class Abra {

int a = 2, b = 1

a = b + x

return pide(a)

return a - y \* b

#### print(0 + 0 + cir.cus(1))

```
class Abra {
   int a = 2, b = 1
0   fun cus(int x): int {
1      a = b + x
2      return pide(a)
   }
   fun pide(int y): int {
3      return a - y * b
   }
}
```

	p	o		cir	<u>.</u>
ra	cus	Abra		cus	Abr
ra	pide	Abra		pide	Cao abr
			l		

ho

cus

```
print(0 + 0 + cir.cus(1))
```

```
class Abra {
    int a = 2, b = 1
0    fun cus(int x): int {
1         a = b + x
2         return pide(a)
      }
    fun pide(int y): int {
3         return a - y * b
      }
}
```

po

Abra

Abra

cus

pide

ho

Abra

Abra

cus

cir		
cus	Abra	
pide	Cad abra	

	х	1
cus@Abra	b	1
	а	2

```
print(0 + 0 + cir.cus(1))
```

```
class Abra {
    int a = 2, b = 1
0    fun cus(int x): int {
1         a = b + x
2         return pide(a)
      }
    fun pide(int y): int {
3         return a - y * b
      }
}
```

po

Abra

Abra

cus

pide

ho

Abra

Abra

cus

cir		
cus	Abra	
pide	Cad abra	

	Х	1
cus@Abra	b	1
	а	2⁄2

```
print(0 + 0 + cir.cus(1))
```

po

Abra

Abra

cus

pide

ho

Abra

Abra

cus

cir					
cus	Abra				
pide	Cad abra				

cus@Abra	Х	1
	b	1
	а	2⁄2

print(
$$0 + 0 + cir.cus(1)$$
)

```
class Cadabra extends Abra {
   Abra zo = new PataDeCabra()
0   fun pide(int y): int {
1     return zo.cus(a + b) - y
   }
}
```

h	o	р	o	ci	r	ZO	)
cus	Abra	cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra	pide	Abra
					abia		

	у	2
pide@Cad	zo	Tipo Abra
abra	b	1
	а	2′2
	х	1
cus@Abra	b	1
	а	2⁄2

print(
$$0 + 0 + cir.cus(1)$$
)

```
class Cadabra extends Abra {
   Abra zo = new PataDeCabra()
0   fun pide(int y): int {
1     return zo.cus(a + b) - y
   }
}
```

h	O	p	O		ciı	r		ZC	)
cus	Abra	cus	Abra	cu	s	Abra		cus	Abra
pide	Abra	pide	Abra	pic	de	Cad abra		pide	Abra
						abia	'		

	у	2
pide@Cad	zo	Tipo Abra
abra	b	1
	а	2′2
	х	1
cus@Abra	b	1
	а	2⁄2

#### print(0+0+cir.cus(1))

```
class Abra {
      int a = 2, b = 1
        fun cus(int x): int {
           a = b + x
           return pide(a)
       fun pide(int y): int {
           return a - y * b
   ho
                             cir
                 po
                                            ZO
                                                Abra
cus
      Abra
             cus
                   Abra
                           cus
                                 Abra
                                          cus
                           pide
pide
      Abra
             pide
                   Abra
                                 Cad
                                          pide
                                                Abra
                                 abra
```

	х	3
cus@Abra	b	1
	а	2
	у	2
pide@Cad	zo	Tipo Abra
abra	b	1
	а	2′2
	х	1
cus@Abra	b	1
	а	2⁄2

#### print(0 + 0 + cir.cus(1))

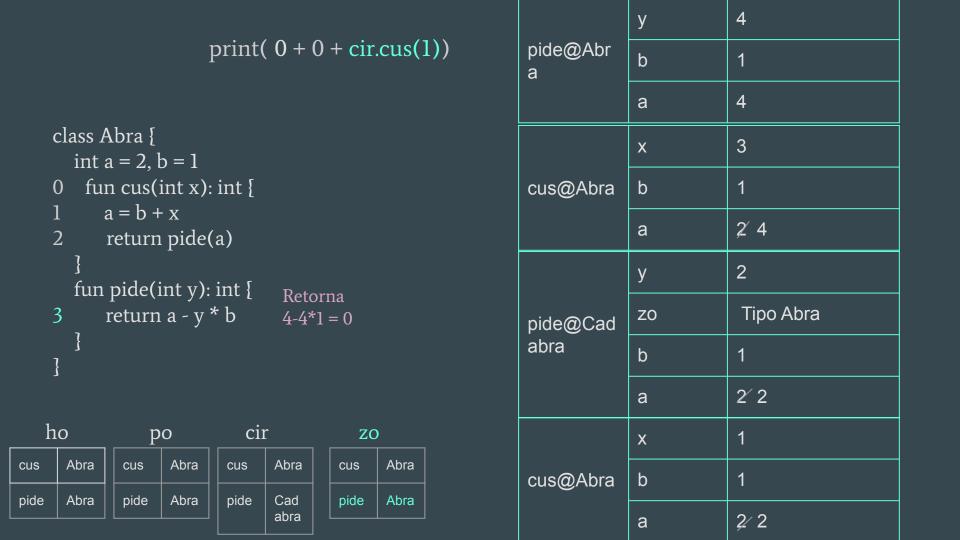
```
class Abra {
      int a = 2, b = 1
        fun cus(int x): int {
           a = b + x
           return pide(a)
       fun pide(int y): int {
           return a - y * b
   ho
                             cir
                 po
                                            ZO
                                                Abra
cus
      Abra
             cus
                   Abra
                           cus
                                 Abra
                                         cus
                           pide
pide
      Abra
             pide
                   Abra
                                 Cad
                                         pide
                                                Abra
                                 abra
```

	х	3
cus@Abra	b	1
	а	<i>2</i> ′ 4
	у	2
pide@Cad	zo	Tipo Abra
abra	b	1
	а	2′2
	х	1
cus@Abra	b	1
	а	2⁄2

#### print(0+0+cir.cus(1))

```
class Abra {
      int a = 2, b = 1
       fun cus(int x): int {
           a = b + x
           return pide(a)
       fun pide(int y): int {
           return a - y * b
   ho
                             cir
                 po
                                            ZO
                                                Abra
cus
      Abra
             cus
                   Abra
                           cus
                                 Abra
                                         cus
                           pide
pide
      Abra
             pide
                   Abra
                                 Cad
                                         pide
                                                Abra
                                 abra
```

cus@Abra       x       3         b       1         a       2 4         y       2         zo       Tipo Abra         b       1         a       2 2         x       1         cus@Abra       b       1         a       2 2				
a       2′ 4         pide@Cad abra       y       2         b       1         a       2′ 2         x       1         cus@Abra       b       1		cus@Abra	х	3
pide@Cad abra       y       2         zo       Tipo Abra         b       1         a       2′2         x       1         cus@Abra       b       1			b	1
pide@Cad abra  b 1 a 2′2 x 1 cus@Abra b 1			а	2′4
abra b 1  a 2′2  x 1  cus@Abra b 1			у	2
a 2´2  x 1  cus@Abra b 1			zo	Tipo Abra
x 1 cus@Abra b 1			b	1
cus@Abra b 1			а	2′2
		cus@Abra	х	1
a 2⁄2			b	1
			а	2⁄2



$$print(0 + 0 + cir.cus(1))$$

```
class Abra {
      int a = 2, b = 1
        fun cus(int x): int {
           a = b + x
                                 Retorna
           return pide(a)
                                 pide(a) = 0
       fun pide(int y): int {
           return a - y * b
   ho
                             cir
                 po
                                            ZO
cus
      Abra
             cus
                   Abra
                           cus
                                 Abra
                                         cus
                                                Abra
pide
      Abra
             pide
                   Abra
                           pide
                                 Cad
                                         pide
                                                Abra
                                 abra
```

	х	3
cus@Abra	b	1
	а	2′4
	у	2
pide@Cad	zo	Tipo Abra
abra	b	1
	а	2′2
cus@Abra	х	1
	b	1
	а	2⁄2

$$print(0+0+cir.cus(1))$$

```
class Cadabra extends Abra {
   Abra zo = new PataDeCabra()
0   fun pide(int y): int {
1     return zo.cus(a + b) - y
   }
}
zo.cus(4) = 0
```

h	o	po		_	cir	:
cus	Abra	cus	Abra		cus	Abra
pide	Abra	pide	Abra		pide	Cad abra

cus	Abra
pide	Abra

pide@Cad	у	2
	zo	Tipo Abra
abra	b	1
	а	2′2
	х	1
cus@Abra	b	1
	а	2⁄2

$$print(0+0+cir.cus(1))$$

```
class Cadabra extends Abra {
   Abra zo = new PataDeCabra()
0 fun pide(int y): int {
1 return zo.cus(a + b) - y Retorna -2
   }
}
```

h	o	p	ро		cir			ZO	)
cus	Abra	cus	Abra		cus	Abra		cus	Abr
pide	Abra	pide	Abra		pide	Cad abra		pide	Abr
				`		abia	'		

	pide@Cad abra	у	2
pide		zo	Tipo Abra
		b	1
		а	2′ 2
	cus@Abra	х	1
cus		b	1
		а	2⁄2

#### print(0 + 0 + cir.cus(1))

h	0	po		ciı	
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

cus@Abra	х	1
	b	1
	а	2⁄2

print(
$$0 + 0 + cir.cus(1)$$
)

cir.cus(1) = -2

h	0	po		cir		
cus	Abra	cus	Abra		cus	Abra
pide	Abra	pide	Abra		pide	Cad abra

print(
$$0 + 0 + -2$$
)

h	0	po		ciı	<u>.</u>
cus	Abra	cus	Abra	cus	Abra
pide	Abra	pide	Abra	pide	Cad abra

Imprime

Abra ho = new Cadabra()

Abra po = new PataDeCabra()

Cadabra cir = new PataDeCabra()

ho

Abra

Abra

cus

pide

cus

pide

print(ho.cus(1) + po.cus(1) + cir.cus(1))

)	cir	-
Abra	cus	Abra
Abra	pide	Cad abra

# guerrero." — Cell

"Todavía no has visto todo mi poder, joven