Code

```
from itertools import count
from toolz import take, cons

def sieve(numbers):
    p = next(numbers)
    yield from cons(p, sieve(n for n in numbers if n % p > 0))

primes = take(10, sieve(count(2)))
for p in primes:
    print(p)
```

Bytecode

```
  line     byte opname                arg argval
  1           0 LOAD_CONST               0 (0)
              2 LOAD_CONST               1 (('count',))
              4 IMPORT_NAME              0 (itertools)
              6 IMPORT_FROM              1 (count)
              8 STORE_NAME               1 (count)
             10 POP_TOP

  2          12 LOAD_CONST               0 (0)
             14 LOAD_CONST               2 (('take', 'cons'))
             16 IMPORT_NAME              2 (toolz)
             18 IMPORT_FROM              3 (take)
             20 STORE_NAME               3 (take)
             22 IMPORT_FROM              4 (cons)
             24 STORE_NAME               4 (cons)
             26 POP_TOP

  4          28 LOAD_CONST               3 (<code object sieve at 0x7fdb5d72e240,
             30 LOAD_CONST               4 ('sieve')
             32 MAKE_FUNCTION            0
             34 STORE_NAME               5 (sieve)

  8          36 LOAD_NAME                3 (take)
```

```
              38 LOAD_CONST               5 (10)
              40 LOAD_NAME                5 (sieve)
              42 LOAD_NAME                1 (count)
              44 LOAD_CONST               6 (2)
              46 CALL_FUNCTION            1
              48 CALL_FUNCTION            1
              50 CALL_FUNCTION            2
              52 STORE_NAME               6 (primes)

  9           54 LOAD_NAME                6 (primes)
              56 GET_ITER
        >>    58 FOR_ITER                12 (to 72)
              60 STORE_NAME               7 (p)

 10           62 LOAD_NAME                8 (print)
              64 LOAD_NAME                7 (p)
              66 CALL_FUNCTION            1
              68 POP_TOP
              70 JUMP_ABSOLUTE           58
        >>    72 LOAD_CONST               7 (None)
              74 RETURN_VALUE

Disassembly of <code object sieve at 0x7fdb5d72e240, file "<dis>", line 4>:
  5           0 LOAD_GLOBAL               0 (next)
              2 LOAD_FAST                 0 (numbers)
              4 CALL_FUNCTION             1
              6 STORE_DEREF               0 (p)

  6           8 LOAD_GLOBAL               1 (cons)
             10 LOAD_DEREF                0 (p)
             12 LOAD_GLOBAL               2 (sieve)
             14 LOAD_CLOSURE              0 (p)
             16 BUILD_TUPLE               1
             18 LOAD_CONST                1 (<code object <genexpr> at 0x7fdb5d72e1
             20 LOAD_CONST                2 ('sieve.<locals>.<genexpr>')
             22 MAKE_FUNCTION             8 (closure)
             24 LOAD_FAST                 0 (numbers)
             26 GET_ITER
```

```
              28 CALL_FUNCTION            1
              30 CALL_FUNCTION            1
              32 CALL_FUNCTION            2
              34 GET_YIELD_FROM_ITER
              36 LOAD_CONST               0 (None)
              38 YIELD_FROM
              40 POP_TOP
              42 LOAD_CONST               0 (None)
              44 RETURN_VALUE

Disassembly of <code object <genexpr> at 0x7fdb5d72e190, file "<dis>", line 6>:
  6           0 LOAD_FAST                0 (.0)
        >>    2 FOR_ITER                22 (to 26)
              4 STORE_FAST               1 (n)
              6 LOAD_FAST                1 (n)
              8 LOAD_DEREF               0 (p)
             10 BINARY_MODULO
             12 LOAD_CONST               0 (0)
             14 COMPARE_OP               4 (>)
             16 POP_JUMP_IF_FALSE        2
             18 LOAD_FAST                1 (n)
             20 YIELD_VALUE
             22 POP_TOP
             24 JUMP_ABSOLUTE            2
        >>   26 LOAD_CONST               1 (None)
             28 RETURN_VALUE
```