Listas y Listas de Listas

Clase #19

IIC1103 – Introducción a la Programación

El plan de hoy es...

- Encuesta ½ semestre
  - Clase pasada: 18
  - Hoy: 18

- Listas
- Y Listas de Listas

hoy

- Recordar i1 este jueves
- Se envió formulario
- Recordar publicación tarea 2

Problema #1

- Escribir una función que agregue un elemento a una lista ordenada (la lista debe seguir ordenada después!).
- `Ejemplo:`

lista = [23, 33, 34, 45, 100] #está ordenada

lista = agregar(lista,35)

lista es: [23, 33, 34, **35**, 45, 100]

Solución

```
def agregar(lista, x):
    for i in range(len(lista)):
        if lista[i]>x:
            lista.insert(i,x)
            return lista
    lista.append(x)
    return lista
```

Como append,
pero indica en que posición
insertar el elemento

```
lista = [23, 33, 34, 45, 100]
lista = agregar(lista,35)
print(lista)
```

😱 Solución 😱

```
def agregar(lista, x):
    for i in range(len(lista)):
        if lista[i]>x:
            lista.insert(i,x)
            return
    lista.append(x)
```

```
lista = [23, 33, 34, 45, 100]
agregar(lista,35)
print(lista)
```

Las listas que se reciben como parámetro de una función SI SE PUEDEN MODIFICAR dentro de la función

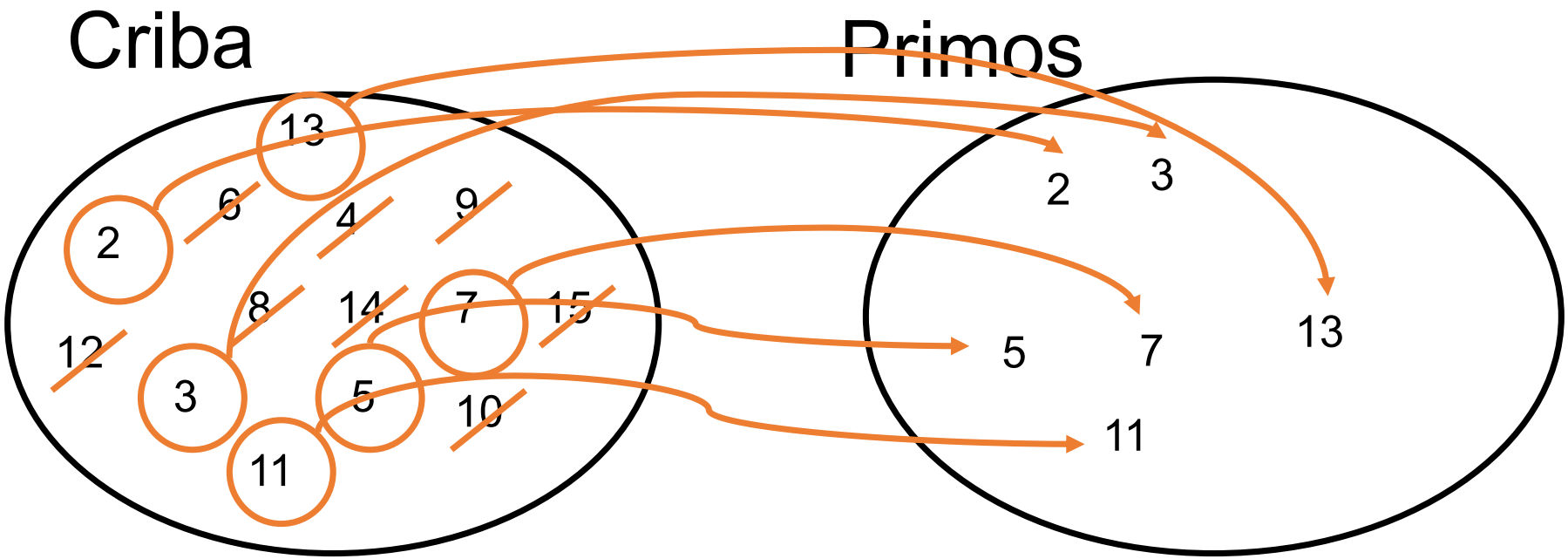
Problema #2

- Encontrar todos los números primos entre 2 y n

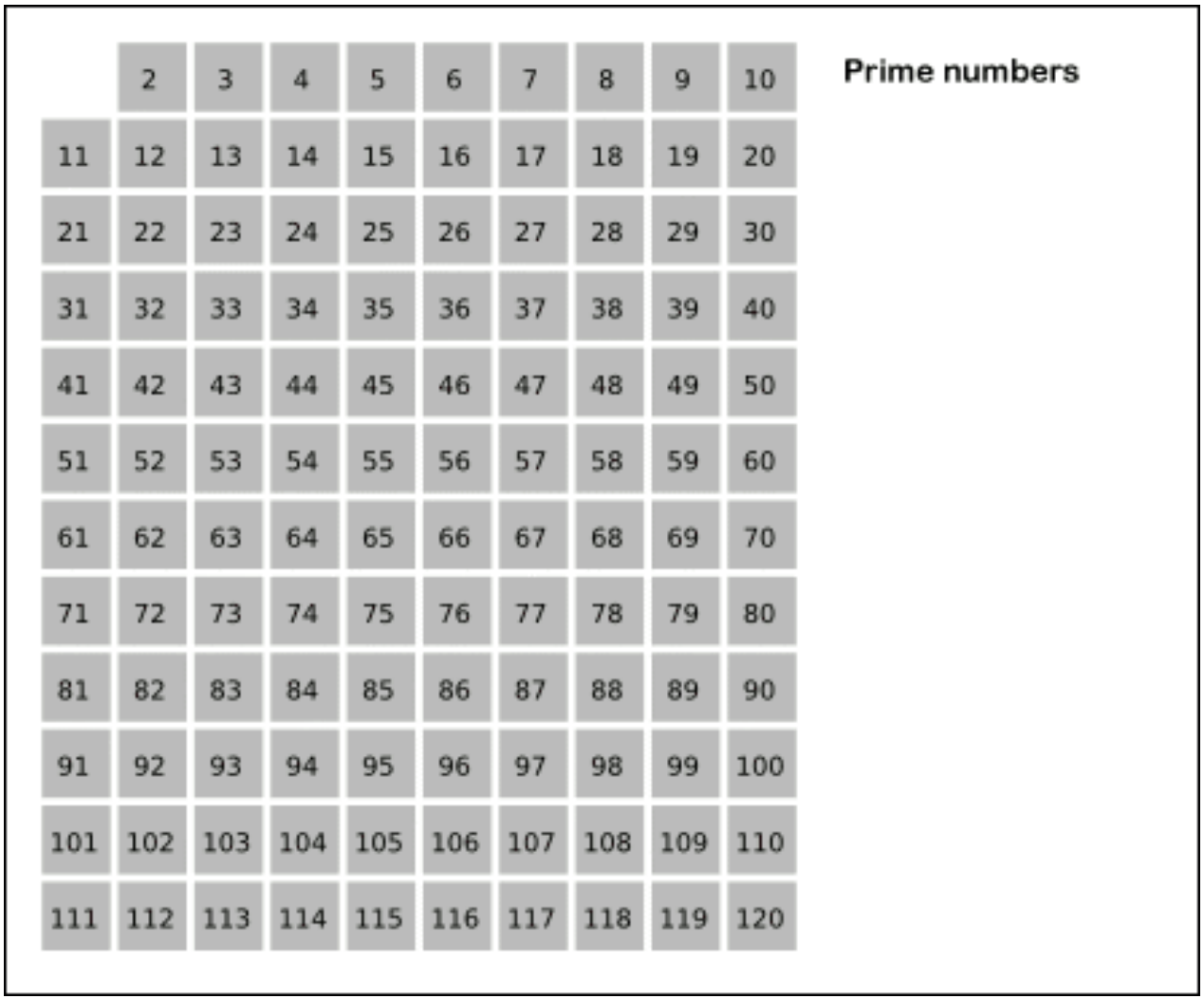
```
n? 10
[2, 3, 5, 7]

n? 40
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37]
```

Algoritmo: Criba de Eratóstenes



Algoritmo: Criba de Eratóstenes hasta 121



	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120

[Fuente: Wikipedia]

Algoritmo

```
n = int(input("n?"))
primos=[]
criba=list(range(2,n+1))
while criba!=[]:
    m=min(criba)
    primos.append(m)
    for i in range(m,n+1,m):
        if i in criba:
            criba.remove(i)
print(primos)
```

Nota:

- lista.remove(valor): elimina 1a aparición del elemento de *valor* de la lista
- lista.pop(indice): elimina el elemento de la posición *índice* y lo retorna

Problema 3: Listas

- Escribir una función eliminar_duplicados(lista) que recibe una lista y elimina todos los números que están más de una vez.

- `lista = [1, 8, 1, 3, 8, 2, 4, 1, 3, 4, 5, 6, 3, 2, 4, 7, 4, 4, 4, 2, 3]`

Solución?? 🙆‍♀️

-
```
def eliminar_duplicados(lista):
    for elem in lista:
        if lista.count(elem)>1:
            while lista.count(elem)>0:
                lista.remove(elem)
```

-
```
lista = [1, 8, 1, 3, 8, 2, 4, 1, 3, 4, 5, 6, 3, 2,
4, 7, 4, 4, 4, 2, 3]
eliminar_duplicados(lista)
print(lista)
```

Solución

Mucho cuidado al modificar una lista mientras la recorremos.

```
def eliminar_duplicados(lista):
    numeros_a_eliminar = []
    for elem in lista:
        if lista.count(elem)>1:
            numeros_a_eliminar.append(elem)

    for elem in numeros_a_eliminar:
        while lista.count(elem)>0:
            lista.remove(elem)

lista = [1, 8, 1, 3, 8, 2, 4, 1, 3, 4, 5, 6, 3, 2,
4, 7, 4, 4, 4, 2, 3]
eliminar_duplicados(lista)
print(lista)
```

¡Quiero más funciones de listas!

Resumen de Listas

- **Hoy vimos**: Dado x: elemento, l,m:lista, s:string, i,j:int
- `l.append(x)–> agrega x al final`
- `l.remove(x) -> elimina x`
- `l.insert(i,x) -> agrega x en posición i`
- `l.pop(i) -> elimina elemento de posición i`
- `x in l: True si x está en l, False si no`

- `s.split(d) -> entrega lista de string s, separado según d`
- `d.join(l) -> entrega string de lista l, unido según d`
- `list(s) -> entrega lista de cada carácter de string s`

- `l[i:j] -> trozo de lista entre i y j-1`
- `len(l) -> nº elementos de l`
- `min(l) -> elemento c/menor valor de l`