

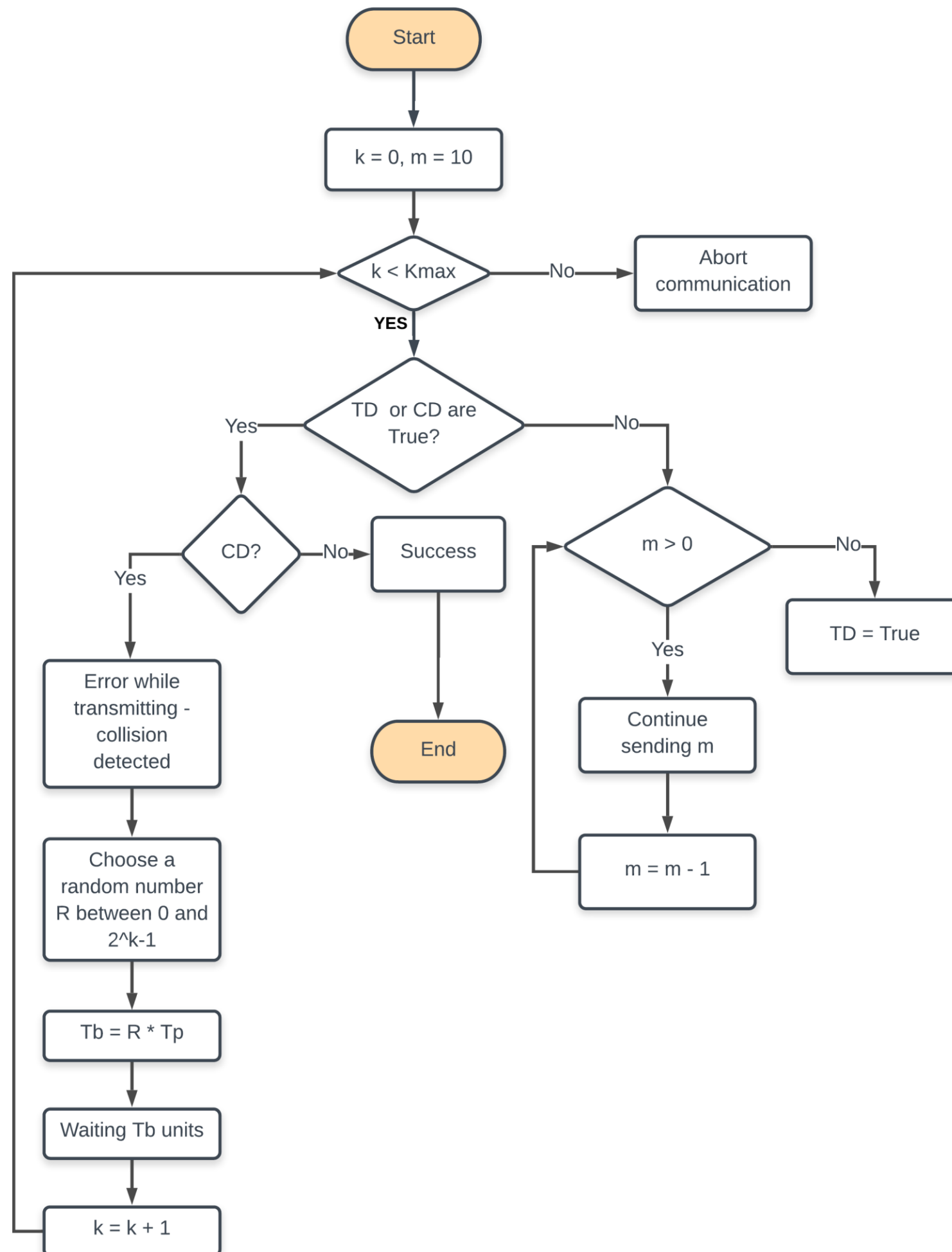
Algorítmica y Programación

Enero - Mayo 2020

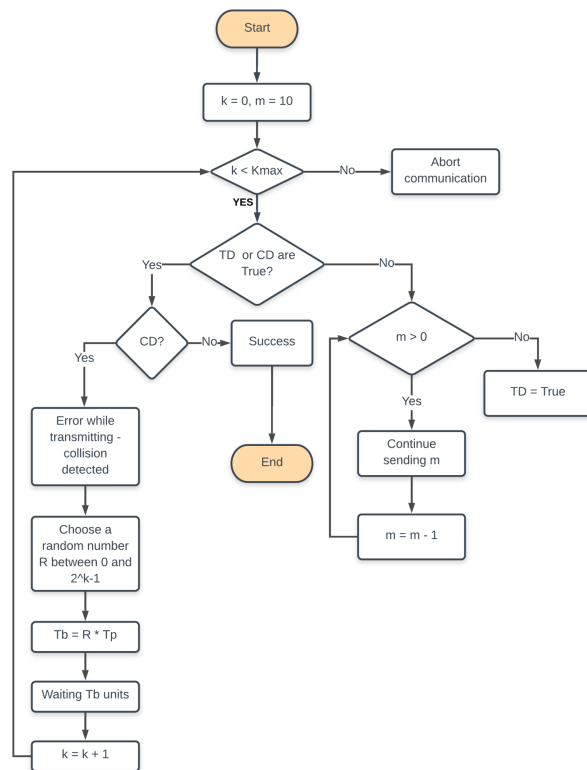
Dr. Iván S. Razo Zapata
(ivan.razo@itam.mx)

Examen

Examen



Examen

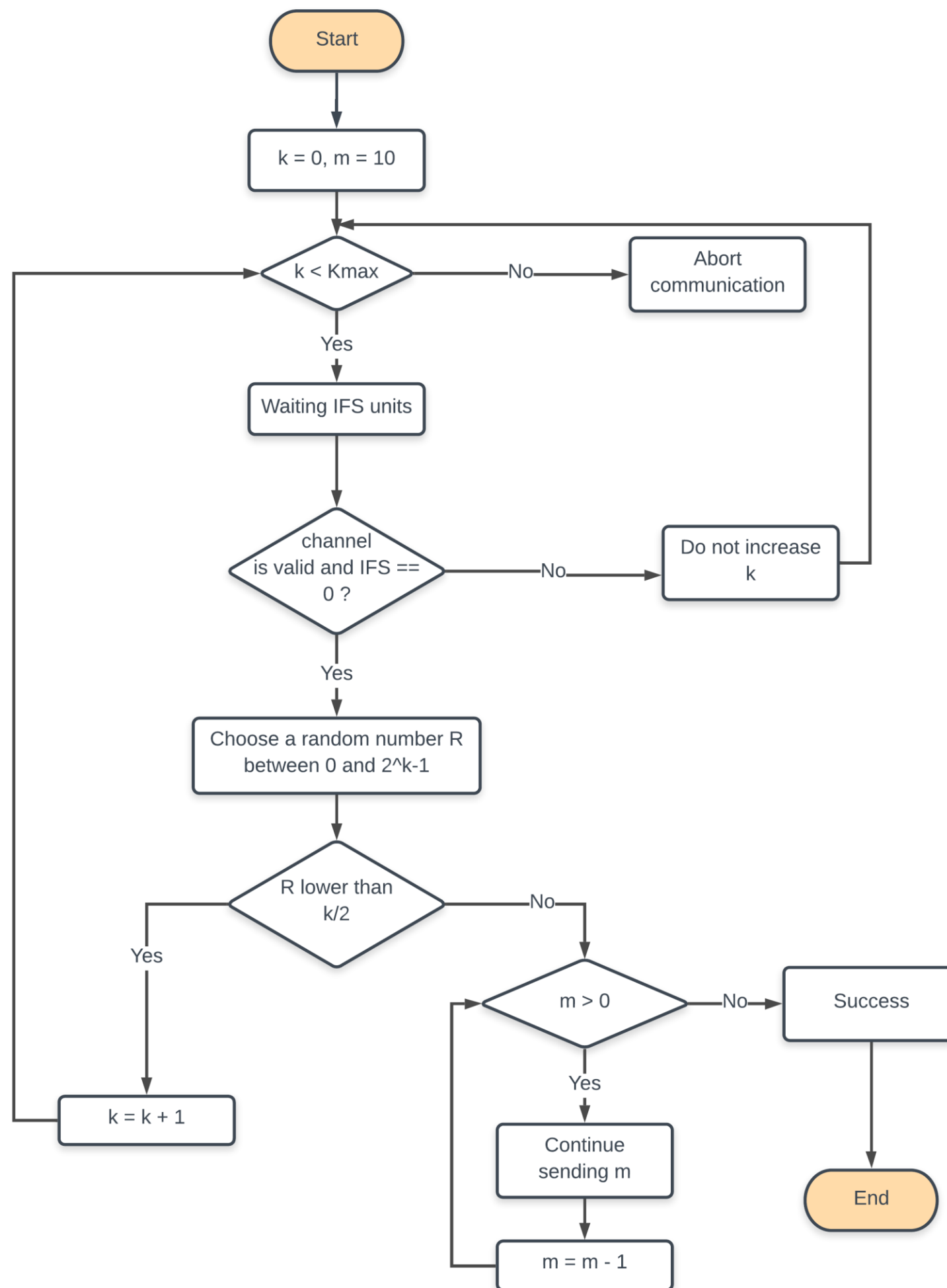


```

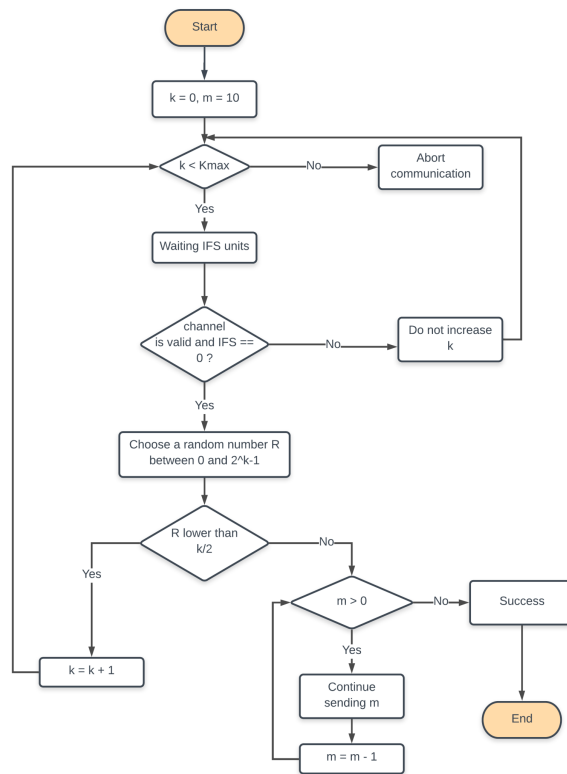
7 import numpy as np
8
9 k= 0
10 m = 10
11 kmax = 20
12 TD = False
13 CD = True # NOTE
14 Tb = 0
15 Tp = 1
16
17 while k < kmax:
18     if (TD or CD) == True:
19         if CD:
20             print("Error while transmitting - cd")
21             R = np.random.uniform(low=0, high=2**(k-1), size=1)
22             Tb = R * Tp
23             print("Waiting Tb units", Tb, R)
24             k = k + 1
25             # CD should change to False at some point
26         else:
27             print("Success")
28             break
29     else:
30         while m > 0:
31             print("continue sending m ")
32             m = m - 1
33         TD = True
34

```

Examen



Examen



```

8 import numpy as np
9
10 k= 0
11 m = 10
12 kmax = 20
13 IFS = 15
14 channel = True
15
16 while k < kmax:
17     print("Waiting IFS units")
18     # Decrease IFS
19     if (channel == True) and (IFS == 0):
20         R = np.random.uniform(low=0, high=2**(k-1), size=1)
21         print("R = ", R)
22         if R < (k/2):
23             k = k + 1
24         else:
25             while m < 0:
26                 print("Sending m ")
27                 m -= 1
28                 print("Successful transmission")
29                 break
30     else:
31         print("Do not increase k")
32     # channel = True or False
  
```

Examen

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

La organización XYZ inc. tiene un registro (archivo “colaboradores.csv”) de colaboradores de la siguiente forma.

Escribe un código en Python que permita leer el contenido del archivo y realice las siguientes operaciones:

Lectura de archivo y almacenamiento de información en diccionario (1 punto):

1. Leer dos parámetros, **N** y **M**
2. Crear un DataFrame (**dataF1**) con el contenido del archivo.
3. Crear una lista con los ids de los usuarios (**listaUsers**), una lista con las edades (**listaAge**) y generar un diccionario (**dict1**) usando ambas listas.
4. Crear series para age (**serieAge**), occupation (**serieOccupation**), y Zip (**serieZip**).

Examen

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

```
8 import pandas as pd
9 import sys
10
11 # Lectura
12 if len(sys.argv) > 1:
13     N = int(sys.argv[1])
14     M = int(sys.argv[2])
15
16 dataF1 = pd.read_csv('colaboradores.csv')
17
18 listaUsers = list(dataF1['userId'])
19 dataF1.set_index('userId', inplace=True)
20 listaAge = list(dataF1['age'])
21
22 dict1 = dict(zip(listaUsers, listaAge))
23
24 serieAge = pd.Series(dataF1['age'])
25 serieOccupation = pd.Series(dataF1['occupation'])
26 serieZip = pd.Series(dataF1['Zip'])
27
```


Examen

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

Procesamiento de información (3 puntos): **3.5**

1. Utilizando comprensión de diccionarios, genere un nuevo diccionario (**dict2**) en donde la edad de los usuarios sea menor o igual a la media de los primeros 20 colaboradores y los ids sean números pares entre N y M
2. Agregar una columna a dataF1 (**new**) cuyo valor será **age** al cuadrado dividida por **occupation** al cubo.
3. Crear una serie (**serie2**) con dict2
4. Crear un nuevo dataframe (**dataF2**) utilizando serie2
5. Agregar una columna a dataF2 (**New**) cuyo valor sea el de la columna **new** al cubo en dataF1 dividido por **age**
6. Agregar una columna a dataF2 (**valZip**) cuyo valor sea el valor de **Zip** en dataF1 multiplicado por el valor de **occupation** en dataF1

Examen

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

```
28 # Procesamiento
29 promedio = dataF1.age.mean()
30 dict2 = {k:v for (k,v) in dict1.items() if ( (k in range(N,M+1)) \
31                                             and ((k % 2) != 0) ) \
32                                             and (v > promedio) )}
33
34 dataF1['new'] = dataF1.age ** 2 / dataF1.occupation ** 3
35
36 serie2 = pd.Series(dict2)
37 dataF2 = pd.DataFrame(serie2)
38
39 dataF2['valNew'] = (dataF1.new ** 3) / (dataF1.age)
40 dataF2['valZip'] = dataF1.Zip * dataF1.occupation
41
```

Examen

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

Impresión de información (2 puntos):

1. Imprimir los valores de serieZip cuyas posiciones en la serie sean mayores a M
2. Imprimir los valores de serieAge cuyas posiciones en la serie estén entre $N/2$ y $M/2$
3. Imprimir los últimos 3 valores y los primeros 6 de dataF2

Exportar resultados (1 punto):

Exportar en un archivo csv (salida.csv) los últimos 20 valores de dataF2 ordenados de manera descendente por el valor de la columna valNew.

Examen

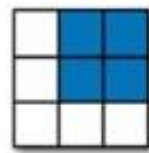
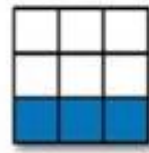
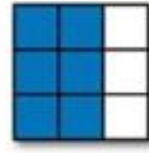

userId	gender	age	occupation	Zip
1	F	25	1	48067
2	M	56	2	70072
3	F	30	5	55117
...				
1000	M	32	6	45027

```

42 # Impresion
43 print(serieZip[:N])
44 print(serieAge[int(N/2):int(M/2)])
45 print(dataF2.tail(3))
46 print(dataF2.head(6))
47
48 # Exportar
49 dataF2.sort_values('valNew', ascending=False).head(20).to_csv('salida.csv')
50

```

Accediendo al contenido

	Expression	Shape
	arr[:2, 1:]	(2, 2)
	arr[2] arr[2, :] arr[2:, :]	(3,) (3,) (1, 3)
	arr[:, :2]	(3, 2)
	arr[1, :2] arr[1:2, :2]	(2,) (1, 2)

Funciones - Intro

Funciones

- Una función es un fragmento de código con un nombre asociado que realiza una serie de tareas y devuelve un valor.
- A los fragmentos de código que tienen un nombre asociado y no devuelven valores se les suele llamar procedimientos.
- En Python no existen los procedimientos, ya que cuando el programador no especifica un valor de retorno la función devuelve el valor None (nada), equivalente al null de Java.

Funciones

- Además de ayudarnos a programar y depurar dividiendo el programa en partes las funciones también permiten reutilizar código
- En Python las funciones se declaran de la siguiente forma:

```
def mi_funcion(param1, param2):  
    print param1  
    print param2
```

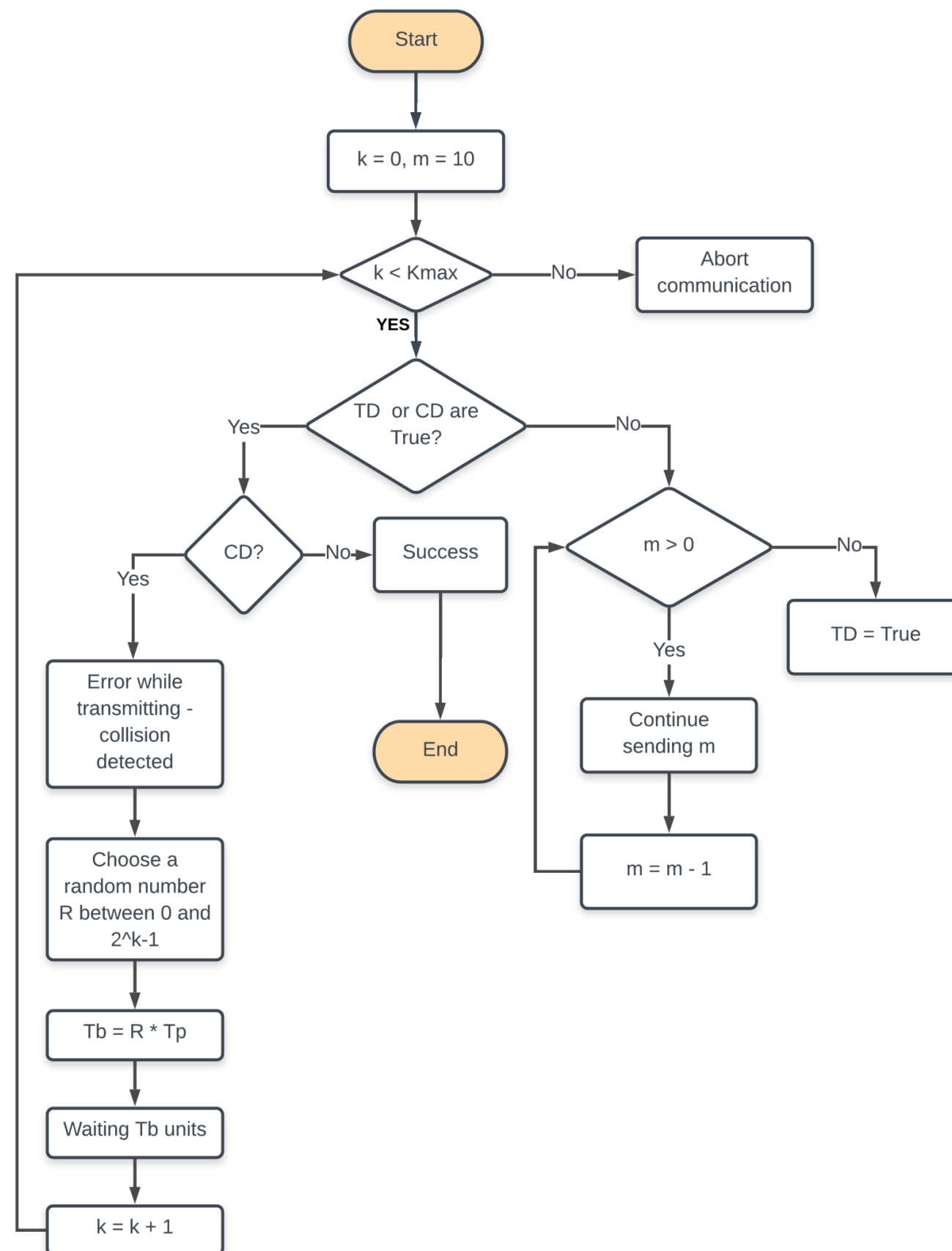
Funciones

```
8 def imprime(N):  
9     while N > 0:  
10         print('Valor de N = ', N)  
11         N -= 1  
12     return N  
13  
14 def divide(M):  
15     while M > 2:  
16         print('Valor de M = ', M)  
17         M = M / 2  
18     return M
```


Funciones

```
20 # Llamado a funciones
21 myN = imprime(23)
22 myM = divide(16)
23
24 print("N ahora vale :", myN)
25 print("M ahora vale :", myM)
26
```

Funciones



Funciones

```
7 import numpy as np
8
9 k= 0
10 m = 10
11 kmax = 20
12 TD = False
13 CD = True # NOTE
14 Tb = 0
15 Tp = 1
16
17 while k < kmax:
18     if (TD or CD) == True:
19         if CD:
20             print("Error while transmitting - cd")
21             R = np.random.uniform(low=0, high=2**(k-1), size=1)
22             Tb = R * Tp
23             print("Waiting Tb units", Tb, R)
24             k = k + 1
25             # CD should change to False at some point
26         else:
27             print("Success")
28             break
29     else:
30         while m > 0:
31             print("continue sending m ")
32             m = m - 1
33             TD = True
34 |
```