mcpp_taller1_andres_ramirez

February 7, 2020

1 Taller 1

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 7-feb-2020 11:59 PM

mcpp_taller1_andres_ramirez [andrese.ramirez@urosario.edu.co]

1.1 Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso. Sugiero una estructura similar a la del repositorio del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller1_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto "[Su nombre acá]" con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:
    1. Descárguelo en PDF. Esto puede implicar instalar LaTex en su computador. Resuélvalo por su cuenta, por favor. Recuerde: Google es su amigo.
    2. Suba los dos archivos (.pdf y .ipynb) a su repositorio en GitHub antes de la fecha y hora límites. Asegúrese de que Daniel sea "colaborador" de su repositorio y de que los dos archivos queden en su repositorio, en la nube (no solo en su computador). No lo deje para última hora. Talleres subidos después de la fecha y hora límites no serán valorados, como tampoco lo serán si son remitidos vía e-mail.

(Todos los ejercicios tienen el mismo valor.)

1.2 Zelle, sección 1.10 (p. 17):

- "Multiple Choice", Ejercicios # 1-10.
- "Programming Exercises", Ejercicio # 1.

1.2.1. Desarrollo ejercicios selección múltiple

1. La respuesta es la b (What can be computed?)
2. La respuesta es la d (recipe)
3. La respuesta es la d (It is not practical to solve)
4. La respuesta es la a (RAM)
5. La respuesta es la b (high-level computer languages)
6. La respuesta es la b (a complete computer command)
7. La respuesta es la c (a compiler is no longer needed after a program is translated)
8. La respuesta es la b (main)
9. La respeusta es la a (They make a program more efficient)
10. La respuesta es la d (parameters)

1.2.2. Desarrollo punto 1 ejercicios de programación

Ejercicio 1

[60]:
```
#(a)#
print("Hello, world!")
```

```
Hello, world!
```

[61]:
```
#b#
print("Hello", "world!")
```

```
Hello world!
```

[63]:
```
#c#
print(3)
```

```
3
```

[64]:
```
#d#
print(3.0)
```

```
3.0
```

[65]:
```
#e#
print(2 + 3)
```

```
5
```

[66]:
```
#f#
print(2.0 + 3.0)
```

```
5.0
```

[67]:
```
#g#
print("2" + "3")
```

```
23
```

[68]:
```
#h#
print("2 + 3 =", 2 + 3)
```

```
2 + 3 = 5
```

[69]:
```
#i#
print(2 * 3)
```

```
6
```

[70]:
```
#j#
print(2 ** 3)
```

```
8
```

[71]:
```
#k#
print(2 / 3)
```

```
0.6666666666666666
```

Ejercicio 2

[85]:
```
#2.1#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .25
0.73125
0.76644140625
0.6981350104385375
0.8218958187902304
0.5708940191969317
0.9553987483642099
0.166186721954413
0.5404179120617926
0.9686289302998042
0.11850901017563877
```

[86]:
```
#2.2#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .54
0.9687599999999998
0.11802984336000057
0.40598531780201574
0.940528834171727
0.2181439504268665
0.6651729525442065
0.8685997934165343
0.44512334994382424
0.9632553577865868
0.13803844657988376
```

[87]:
```
#2.2#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .31416
0.8403076281599999
0.5233438010428494
0.9728747611162002
0.10291889516725675
0.3600737251207511
0.8986404866338258
0.3552344734237516
0.8932674750288959
0.3718287030291472
0.910931262667957
```

[91]:
```
#2.2#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .455668768
0.9673354932905125
0.12323039317256207
0.4213741871472816
0.9508901280576723
0.18212256043853006
0.5809203403295917
0.9494624042316778
0.18713583401089137
0.5932524531957639
0.9410855218945902
```

[92]:
```
#2.2#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .0000009912
3.865676168337984e-06
1.5076078777054407e-05
5.8795820806722146e-05
0.00022929021904689347
0.0008940268156651371
0.003483587373700234
0.013538662771569118
0.0520859327895147
0.19255465474034733
0.6063617027448744
```

[94]:
```
#2.2#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: 0.9999999999999
3.9012126862299474e-13
1.521472947629086e-12
5.933744495744407e-12
2.314160353326587e-11
9.025225377764831e-11
3.5198378970106105e-10
1.372736779350957e-09
5.353673432119548e-09
2.087932627348514e-08
8.142937076640162e-08
```

Ejercicio 3

[95]:
```
#3.1#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 2.0 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: 0.54645
0.495684795
0.4999627580116159
0.4999999972260685
0.5
0.5
0.5
0.5
0.5
0.5
0.5
```

[96]:
```
#3.1#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 2.0 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: 0.99999
1.9999799999908982e-05
3.9998800015817894e-05
7.999440022363038e-05
0.0001599760022391265
0.0003199008198356681
0.0006395969666022731
0.0012783757646451725
0.00255483040099081
0.00509395528926014
0.01013595490326354
```

[97]:
```
#3.1#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 2.0 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .123456789
0.21643042049961894
0.33917658716395416
0.4482716597675335
0.49464835763358833
0.4999427198479641
0.49999999343796836
0.49999999999999983
0.5
0.5
0.5
```

[98]:
```
#3.1#
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 2.0 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .987654321
0.024386526420058022
0.04758364749844367
0.09063888797837505
0.16484695992883724
0.27534487946211517
0.39906015363221686
0.4796222948304966
0.49916949826404955
0.4999986205337332
0.4999999999961941
```

3.1 Comentario

La función al cambiarse el 3.9 por el 2.0 empieza con ciertos valores a repetir números, lo que apriori pareciera indicar que ya no es tan aleatorio el resultado.

Ejercicio 4

[100]:
```
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(20):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .25
0.73125
0.76644140625
0.6981350104385375
0.8218958187902304
0.5708940191969317
0.9553987483642099
0.166186721954413
0.5404179120617926
0.9686289302998042
0.11850901017563877
0.4074120362630336
0.9415671289870646
0.214572035332672
0.6572704202448796
0.8785374581723959
0.4161666317654883
0.9475906688447814
0.19368411333601687
0.6090652525513056
0.9286086056750876
```

Ejercicio 5

[132]:
```
def main():
    print("This program illustrates a chaotic function")
    n = eval(input("¿Cuantos resultados desea que muestre la función? "))
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(n):
        x = 3.9 * x * (1 - x)
        print(x)
main()
```

```
This program illustrates a chaotic function
¿Cuantos resultados desea que muestre la función? 25
Enter a number between 0 and 1: .7587598239015
0.7138690787844124
0.7966140668456365
0.6318783718622398
0.9071715706363627
0.3284241170555183
0.8601906939281039
0.46902438962960485
0.9712579950925077
0.10887201803942112
0.3783737167370358
0.9173074841565849
0.29583240830091634
0.8124308185492055
0.5943092361206262
0.9403124951311415
0.21888713586241526
0.6668046747038189
0.8664871819381174
0.4511798673527233
0.965704719128379
0.12916454686807488
0.4386761601317151
0.960333607988855
0.14856258048825624
0.4933177866510139
```

Ejercicio 6

[147]:
```
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    y = eval(input("Enter a number between 0 and 1: "))
    z = eval(input("Enter a number between 0 and 1: "))
    for i in range(100):
        x = 3.9 * x * (1 - x)
        y = 3.9 * (y - y * y)
        z = 3.9 * x - 3.9 * x * x
        print(x,"         ", y, "     ", z)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .91
Enter a number between 0 and 1: .91
Enter a number between 0 and 1: .91
0.31940999999999986          0.31940999999999986       0.8478102824099998
0.8478102824099998           0.8478102824099999       0.5032092290545171
0.5032092290545171           0.5032092290545167       0.974959833310615
0.974959833310615           0.9749598333106151       0.09521131129205607
0.09521131129205605          0.09521131129205575       0.3359698582270027
0.3359698582270027           0.3359698582270017       0.8700670391007286
0.8700670391007285           0.8700670391007272       0.44089650762775756
0.4408965076277576           0.4408965076277611       0.9613764310386768
0.9613764310386769           0.9613764310386783       0.14481397663985085
0.14481397663985088          0.14481397663984577       0.482987266357454
0.482987266357454           0.48298726635743994       0.9738712108866303
0.9738712108866304           0.9738712108866284       0.09923969442207081
0.09923969442207069          0.09923969442207796       0.3486255921450304
0.3486255921450304           0.3486255921450532       0.8856345757215736
0.8856345757215736           0.8856345757216003       0.39501529863136353
0.39501529863136364          0.3950152986312831        0.932015028666299
0.932015028666299           0.9320150286662332       0.24711575852518175
0.24711575852518158          0.24711575852540366       0.7255932856134524
0.7255932856134524           0.7255932856138904       0.7765199109958965
0.7765199109958965           0.776519910995126        0.6767932814090036
0.6767932814090039           0.6767932814106656       0.8531021290296834
0.8531021290296834           0.8531021290273918       0.48874365725134883
0.4887436572513488           0.48874365725766034       0.974505849516907
0.9745058495169069           0.9745058495174611       0.09689237521953009
0.09689237521953022          0.09689237521747905       0.3412665470910071
0.3412665470910071           0.3412665470845577       0.8767343946175952
0.8767343946175953           0.8767343946096102       0.4214776640572446
0.42147766405724457          0.421477664080709        0.9509535467565629
0.9509535467565627           0.9509535467709344       0.18189950480393868
0.18189950480393866          0.1818995047533876        0.5803670923284802
```

0.5803670923284802 0.5803670922030537 0.9498104088355956
0.9498104088355956 0.9498104089142209 0.18591532480230155
0.18591532480230175 0.1859153245264435 0.5902681855432299
0.5902681855432299 0.5902681848674157 0.9432214532470586
0.9432214532470586 0.943221453722894 0.2088634991881051
0.2088634991881051 0.20886349754308145 0.6444341977905215
0.6444341977905215 0.6444341940548953 0.8936411737835732
0.8936411737835732 0.8936411779920801 0.37068184257893355
0.3706818425789338 0.3706818296571295 0.9097795752287592
0.9097795752287592 0.909779562194772 0.3201147289288171
0.320114728928817 0.32011477058909793 0.8488010280814233
0.8488010280814233 0.8488010865351697 0.5005175869564344
0.5005175869564343 0.5005174279243517 0.9749989552045958
0.9749989552045958 0.9749989558465381 0.09506637096271531
0.09506637096271539 0.09506636858432468 0.33551214869130963
0.33551214869130963 0.33551214117920464 0.8694806124102157
0.8694806124102156 0.8694806027721453 0.44258790050659114
0.442587900506591 0.44258792828301524 0.9621450182438597
0.9621450182438597 0.9621450306825388 0.14204573023829203
0.14204573023829217 0.1420456854001976 0.4752880889615137
0.4752880889615137 0.4752879637716042 0.9726183536660211
0.9726183536660212 0.9726183295352411 0.10386437793427694
0.10386437793427675 0.10386446689054131 0.36299861882934964
0.36299861882934964 0.3629988936915313 0.9017994240736031
0.9017994240736031 0.901799717793997 0.3453731689750712
0.3453731689750712 0.345372248444591 0.8817531181960336
0.8817531181960336 0.881752007950783 0.4066317713156602
0.40663177131566014 0.4066350772596268 0.9410012581021604
0.9410012581021603 0.9410036656865681 0.2165197723740162
0.21651977237401604 0.21651149071893772 0.6615919461259692
0.6615919461259693 0.6615736339118441 0.8731633674941652
0.8731633674941651 0.8731864472156334 0.43192149452561734
0.4319214945256172 0.4318543148968355 0.9569247366602602
0.9569247366602603 0.9568890458470993 0.16075716160901177
0.16075716160901177 0.16088435916121577 0.5261657567416608
0.5261657567416608 0.5265022703401527 0.9723298773791313
0.9723298773791313 0.972260755700588 0.1049274990465574
0.1049274990465575 0.10518213663792712 0.3662791040625295
0.36627910406252945 0.36706353360380944 0.9052630157597527
0.9052630157597525 0.9060787940181685 0.334471363423511
0.3344713634235112 0.3318900508901247 0.8681410548452019
0.8681410548452018 0.864782275540073 0.4464414385761022
0.44644143857610236 0.4560421766620442 0.9638127739429901
0.96381277394299 0.9674640680928174 0.13602307183650453
0.13602307183650442 0.12276164566424361 0.45833110348219575
0.4583311034821958 0.4199957736703962 0.9682284419456552
0.9682284419456553 0.950037362700666 0.11997229199726211
0.1199722919972621 0.18511885147637386 0.41175787048572454

```
0.41175787048572454    0.5883144629873241    0.9446319736572647
0.9446319736572647     0.9445821669463161    0.2039793912064174
0.2039793912064173     0.20415211764993996   0.633250016760959
0.633250016760959      0.6336487189850416    0.9057532888294961
0.9057532888294961     0.9053382776632638    0.3329206475557851
0.3329206475557852     0.33423343457781735   0.8661295109486052
0.8661295109486052     0.8678336385737196    0.4522018067288869
0.45220180672888705    0.4473238159011108    0.9660897976080676
0.9660897976080675     0.9641783565522417    0.12776517220611572
0.12776517220611577    0.1346999679029938    0.4346208086113037
0.4346208086113037     0.45456795754471235   0.9583296892001049
0.9583296892001049     0.9669501251215297    0.15574219439116765
0.1557421943911676     0.12463456453105239   0.5127975967818263
0.5127975967818263     0.42549308043530853   0.9743612639147781
0.9743612639147782     0.9533500038542243    0.09742742605897936
0.09742742605897926    0.17344771862096162   0.34294775857095544
0.34294775857095544    0.5591180693519668    0.8788049145022412
0.8788049145022411     0.9613697101167746    0.41537666332090417
0.41537666332090434    0.14483816328837715   0.947071674468256
0.9470716744682559     0.4830542720001435    0.19549497976378438
0.19549497976378427    0.973880084979964     0.6133791013386732
0.6133791013386732     0.09920689373159      0.924866199580576
```

Los números de cada resultado tienden a presentar variaciones muy pequeñas cuando se hace la comparación. Esto puede suceder a priori porque al ejecutar la función la parte entera o número al que se aproxima la salida cambia a pesar de que sean expresiones equivalentes. En otros términos, me atrevería a decir que a pesar de que son funciones equivalentes a la hora de aproximar a un número esto supone un pequeño cambio.

Ejercicio 7

[142]:
```
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    n = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        n = 3.9 * n * (1 - n)
        print(x,"         ", n)
main()
```

```
This program illustrates a chaotic function
Enter a number between 0 and 1: .5647382910
Enter a number between 0 and 1: .1092384756
0.9586549193457573           0.379491181089494
0.15457909334375375           0.9183627358026999
0.5096691492546296           0.2923932230375533
0.9746353794554976           0.8069077620211332
0.09641290061962962           0.6076497398835998
0.3397580675335823           0.9298049806616738
0.8748578400097693           0.25454394653383094
0.42697823915451977           0.7400301706855713
0.9542045075277993           0.7503035169265565
0.17042323483157437           0.7306577827153684
```

En *computer science* son comunes los ejercicios denominados "pensar como un computador". Con estos usted evalúa si está comprendiendo el material, siempre y cuando no utilice un computador para correr el código del enunciado. Siempre que vea un ejercicio marcado con la etiqueta "pensar como un computador", use papel y lápiz o incluso una calculadora si es necesario para descifrar la respuesta, pero nunca ejecute el código en computador.

1.3 [Pensar como un computador] ¿Cuál es el valor de w después de ejecutar el siguiente código?

x = 7

y = 5.0

z = 10.0

w = x % 2 + y / z + z + y / (z + z)

Desarrollo

Si cambiamos los términos podemos plantear la siguiente expresión:

w = 7 % 2 + 5.0/10.0 + 10.0 + 5.0/20.0

w = 1 + 1/2 + 10.0 + 1/4

w = 1 + 0.2 + 10.0 + 0.25

w = 11.75

Si lo corremos mediante un código arroja los siguientes resultados, que coinciden con el desarrollo operacional mostrado en la línea de arriba.

[13]:
```
x = 7
y = 5.0
z = 10.0
```

[14]:
```
x%2 + y/z + z + y/(z+z)
```

[14]:
```
11.75
```

1.4 [Pensar como un computador] ¿Cuál es el valor de *c* después de ejecutar el siguiente código?

c = True

d = False

c = c and d

c = not c or d

Si se corre el codigo, el valor de c seria falso. Si observaramos solo la tercera línea, puesto que es una conjunción, esta solo puede ser cierta si tanto c como d son verdaderas, no obstante, al ser la segunda (d) falsa, ello lleva a que la conjunción (and) sea falsa. Ahora bien, frente a la cuarta línea, al ser esta la negación de una disyunción, la conclusión solo será verdadera cuando las dos premisas sean falsas, cuestión que nunca sucederá. Así entonces, al ser c verdadero pero d falso, por ende, la expresión c or d, será verdadera siempre. Sin embargo, como c en esa línea es igual a la negación de la conjunción -not (c or d)- por ende la expresión será falsa. En síntesis, el valor de c será falso.

[17]:
```
#si corremos el código se corrobora lo expresado en la línea de arriba#
c = True
d = False
c and d
not c or d
```

[17]:
```
False
```

1.5 Ejecute el siguiente código y responda: ¿Por qué es falsa la tercera línea, mientras que las primeras dos son verdaderas?

[21]:
```
1 == 1
"1" == "1"
1 == "1"
```

[21]:
```
False
```

La tercera línea es falsa porque se está igualando un número entero (int) a una variable string. Si bien el 1 es un número entero, al colocarse este entre comillas, se le está indicando al sistema que es una variable de texto. Por ende, si se iguala una variable int con una str, esto para la maquina no tendrá sentido, puesto que no serian equivalentes como variables, ya que no pertenecen al mismo dominio. Una pertenece a los número reales, mientras que la otra a un conjunto completamente distinto con que el que no tiene ningún elemento en común.

1.6 Escriba un programa que le pida al usuario ingresar su nombre y que arroje un texto saludando de vuelta al usuario, así: "Hola, <nombre>. ¡Veo que aprendes Python rápidamente! ¡Felicitaciones!".

[56]:
```
nombre = input("Por favor escriba su nombre: ")
print (f"Hola, {nombre}. ¡Veo que aprendes Python rápidamente! ¡Felicitaciones!")
```

```
Por favor escriba su nombre: Andrés
Hola, Andrés. ¡Veo que aprendes Python rápidamente! ¡Felicitaciones!
```

Como no sabia si después del nombre debia poner un punto, a pesar de saber que tras un signo de interrogación o exclamación no se coloca punto, decidi correr el código de arriba utilizando una cadena "f". Sin embargo, abajo corro el código sin la cadena.

[58]:
```
nombre = input("Por favor escriba su nombre: ")
print ("Hola,", nombre , "¡Veo que aprendes Python rápidamente! ¡Felicitaciones!
 ↪")
```

```
Por favor escriba su nombre: Andrés
Hola, Andrés ¡Veo que aprendes Python rápidamente! ¡Felicitaciones!
```
