



Instituto Tecnológico de Costa Rica
Escuela de Computación
Ingeniería en Computación
IC-6831 Aseguramiento de la calidad del software

DESARROLLO DE PROGRAMAS ALREDEDOR DEL
CALENDARIO GREGORIANO

Asignación 1b

Profesor: Ignacio Trejos Zelaya

Estudiantes:
2016000389- Joshua Arcia López
2018093728 - Paula Mariana Bustos Vargas

Fecha: 8 de Abril, 2022
Semestre I 2022

Índice

1. Cambios respecto a la entrega 1a	2
1.1. Modificaciones realizadas	2
1.2. Historial de versiones	2
1.2.1. V0	2
1.2.2. Diferencias entre componentes	2
1.2.3. V1	2
1.2.4. Diferencias entre componentes	3
1.2.5. V2	3
1.2.6. Diferencias entre componentes	4
1.2.7. V3	4
1.2.8. Diferencias entre componentes	5
1.2.9. V4	6
1.2.10. Diferencias entre componentes	6
1.2.11. V5	6
1.2.12. Diferencias entre componentes	7
1.2.13. V6	8
1.2.14. Diferencias entre componentes	8
2. Asignacion 1b pruebas	9
2.1. R6 (dia_semana)	9
2.2. R7 (fecha_futura)	9
2.3. R8 (dias_entre)	10
2.4. R9 (edad_al)	12
2.5. R10 (fecha_hoy)	12
2.6. R11 (edad_hoy):	13
3. Desglose de tiempo	14
3.1. Mariana	14
3.2. Joshua	14
3.3. Joshua y Mariana	14
4. Código	15
4.1. R6 (dia_semana)	15
4.2. R7 (fecha_futura)	15
4.3. R8 (dias_entre)	16
4.4. R9 (edad_al)	17
4.5. R10 (fecha_hoy)	18
4.6. R11 (edad_hoy):	18
4.7. Código base	19

1. Cambios respecto a la entrega 1a

1.1. Modificaciones realizadas

- Se cambiaron los nombres de las funciones R0 a R5, por los nombres correctos (los que estaban dentro del paréntesis)
- La función día_siguiente ahora cambia de año, si el día es 31 de Diciembre.
- La función día_siguiente ahora cambia de día, si el día se encuentra dentro del mes de Febrero, independientemente si este el año es bisiesto.
- La función día_siguiente ahora tiene un número de identaciones y espacio consistentes entre sí.

1.2. Historial de versiones

1.2.1. V0

La versión 0 consta de las primeras 5 funciones las cuales se había puesto por error el nombre de ellas R# dependiendo de la función que fuese y al parámetro el nombre que tenía que ser de la función, exceptuando la función que validaba si era bisiesto o no.

1.2.2. Diferencias entre componentes

Es la versión original, a continuación se muestran los nombre que se le pusieron a las funciones, mas la función auxiliar que se creo.

- R0(fecha_es_tupla):
- R1(fecha_es_valida):
- R2(fecha_es_valida):
- R3(dia_siguiente):
- R4(ordinal_dia):
- R5(Imprimir_3x4):
- cal_G(dia, mes, anno):

1.2.3. V1

- Se agregaron dos funciones: dias_entre y fechas_hoy.
En el caso de dias_entre, se agregó una versión inicial en la cual faltaba el manejo de años, mientras que fechas_hoy únicamente contaba con la definición de función.

1.2.4. Diferencias entre componentes

<pre>946 947 #RB ----- dias_entre 948 ----- 949 #Lista que tiene 2 tuplas o 2 parametros diferentes</pre>	<pre>946 947 #RB ----- dias_entre 948 ----- 949 #Lista que tiene 2 tuplas o 2 parametros diferentes 950 + #04/06/2022: 1:15 horas, 30 programando, 30 diseñando, 15 documentación 951 + def dias_entre(fecha1, fecha2): 952 + if (fecha_es_tupla(fecha1) and fecha_es_tupla(fecha2) and 953 + fecha_es_valida(fecha1) and fecha_es_valida(fecha2) and fecha1[0] > 1582 and 954 + fecha2[0] > 1582): #Validación con RB y R2 955 + if (fecha1 == fecha2): 956 + return 0 957 + elif (fecha1[0] == fecha2[0]): 958 + if (fecha1[1] == fecha2[1]): 959 + if (fecha1[2] > fecha2[2]): 960 + return fecha1[2] - fecha2[2] 961 + else: 962 + return fecha2[2] - fecha1[2] 963 + else: 964 + if (fecha1[1] > fecha2[1]): 965 + return ordinal_dia(fecha1) - ordinal_dia(fecha2) 966 + else: 967 + return ordinal_dia(fecha2) - ordinal_dia(fecha1) 968 + else: #falta manejar años 969 + pass 970 + else: 971 + print("La fecha no es válida")</pre>
<pre>956 - 957 - #R11 ----- edad_hoy 958 -----</pre>	<pre>976 + def fecha_hoy(): 977 + pass 978 + #R11 ----- edad_hoy 979 -----</pre>

1.2.5. V2

- Se modificó la función dia_siguiente, ya que contaba con errores con en la implementación. La función no validaba que las fechas insertadas fueran acordes a los requerimientos, además, se corrigieron 2 errores: En el mes de febrero, no se devolvían los valores correspondientes al día siguiente, a menos que fuera un año bisiesto, y en el mes de Diciembre, si el día era 31 de Diciembre, la función no devolvía la fecha correcta, devolvía la misma fecha ingresada. Además, se completó la función dias_entre.

1.2.6. Diferencias entre componentes

<pre>@@ -94,36 +94,39 @@ def fecha_es_valida(fecha): 94 95 #R3 ----- dia siguiente 96 ----- 97 #Joshua 97 - def dia_siguiente(dia_actual): #Calcula el día siguiente, dado una fecha. Puede mejorarse 98 - Dias_31 = [1,3,5,7,8,10,12] 99 - Dias_30 = [4,6,9,11] 100 - if (dia_siguiente(dia_actual) == True): 101 - esBisiesto = bisiesto(dia_actual[0]) 102 - dia = dia_actual[2] 103 - mes = dia_actual[1] 104 - if (mes in Dias_31) and (dia == 31): 105 - fecha = (dia_actual[0], dia_actual[1]+1, 1) 106 - return dia_actual 107 - if (mes in Dias_30) and (dia == 30): 108 - fecha = (dia_actual[0], dia_actual[1]+1, 1) 109 - return fecha 110 - if (esBisiesto) and (mes == 2): 111 - if (dia == 29): 112 - dia_actual = (dia_actual[0], dia_actual[1]+1, 1) 113 - return dia_actual 114 - else: 115 - dia_actual = (dia_actual[0], dia_actual[1], dia_actual[2]+1) 116 - return dia_actual 117 - elif (mes == 2) and (dia == 28): 118 - fecha = (dia_actual[0], dia_actual[1]+1, 1) 119 - return dia_actual 120 - else: 121 - dia_actual = (dia_actual[0], dia_actual[1], dia_actual[2]+1) 122 - return dia_actual 123 - #01,03,05,07,08,10,12 124 - #04,06,09,11 125 - else: 126 - print("La fecha ingresada no es válida") 127 128 #R4 ----- ordinal 129 ----- 130 #mari</pre>	<pre>94 95 #R3 ----- dia siguiente 96 ----- 97 #Joshua 97 + def dia_siguiente(dia_actual): 98 + Dias_31 = [1,3,5,7,8,10,12] 99 + Dias_30 = [4,6,9,11] 100 + if (fecha_es_tupla(dia_actual) and fecha_es_valida(dia_actual) and dia_actual[0] > 1582): 101 + esBisiesto = bisiesto(dia_actual[0]) 102 + dia = dia_actual[2] 103 + mes = dia_actual[1] 104 + if (mes in Dias_31) and (dia == 31): 105 + if (mes == 12): #Cambia de año 106 + dia_actual = (dia_actual[0]+1, 1, 1) 107 + return dia_actual 108 + else: 109 + dia_actual = (dia_actual[0], dia_actual[1]+1, 1) 110 + return dia_actual 111 + elif (mes in Dias_30) and (dia == 30): 112 + dia_actual = (dia_actual[0], dia_actual[1]+1, 1) 113 + return dia_actual 114 + elif (esBisiesto) and (mes == 2): 115 + if (dia == 29): 116 + dia_actual = (dia_actual[0], dia_actual[1]+1, 1) 117 + return dia_actual 118 + else: 119 + dia_actual = (dia_actual[0], dia_actual[1], dia_actual[2]+1) 120 + return dia_actual 121 + elif (mes == 2) and (dia == 28): 122 + dia_actual = (dia_actual[0], dia_actual[1]+1, 1) 123 + return dia_actual 124 + else: 125 + dia_actual = (dia_actual[0], dia_actual[1], dia_actual[2]+1) 126 + return dia_actual 127 + else: 128 + print("La fecha ingresada no es válida") 129 130 #R4 ----- ordinal 131 ----- 132 #mari</pre>
<pre>@@ -947,6 +950,7 @@ def dia_semana(fecha): 947 948 #R8 ----- dias_entre 949 ----- 950 #Lista que tiene 2 tuplas o 2 parametros diferentes 951 #04/06/2022: 1:15 horas, 30 programando, 30 diseñando, 15 documentación 952 953 def dias_entre(fecha1, fecha2): 954 if (fecha_es_tupla(fecha1) and fecha_es_tupla(fecha2) and fecha_es_valida(fecha1) and fecha_es_valida(fecha2) and fecha1[0] > 1582 and fecha2[0] > 1582): #Validación con R8 y R2 955 if (fecha1 == fecha2): 956 return ordinal_dia(fecha1) - ordinal_dia(fecha2) 957 else: 958 return ordinal_dia(fecha2) - ordinal_dia(fecha1) 959 else: #Falta manejar años 960 pass 961 962 else: 963 print("La fecha no es válida") 964 965 966</pre>	<pre>950 951 #R8 ----- dias_entre 952 ----- 953 #Lista que tiene 2 tuplas o 2 parametros diferentes 954 #04/06/2022: 1:15 horas, 30 programando, 30 diseñando, 15 documentación 955 + #04/07/2022: 1 hora programando 956 def dias_entre(fecha1, fecha2): 957 if (fecha_es_tupla(fecha1) and fecha_es_tupla(fecha2) and fecha_es_valida(fecha1) and fecha_es_valida(fecha2) and fecha1[0] > 1582 and fecha2[0] > 1582): #Validación con R8 y R2 958 if (fecha1 == fecha2): 959 return ordinal_dia(fecha1) - ordinal_dia(fecha2) 960 else: 961 return ordinal_dia(fecha2) - ordinal_dia(fecha1) 962 else: 963 dias = 0 964 if (fecha1[0] > fecha2[0]): 965 while (fecha1 > fecha2): 966 fecha2 = dia_siguiente(fecha2) 967 dias += 1 968 return dias 969 else: 970 while (fecha2 > fecha1): 971 fecha1 = dia_siguiente(fecha1) 972 dias += 1 973 return dias 974 975 else: 976 print("La fecha no es válida") 977 978 979</pre>

1.2.7. V3

- Se agregaron varios comentarios descriptivos, así como la función `fecha_hoy` y algunas pruebas.

1.2.8. Diferencias entre componentes

[illegible]

1.2.9. V4

- Se agregó la función `edad_hoy`.

1.2.10. Diferencias entre componentes

```
.
r11
main
mari committed yesterday
1 parent 5c496bc commit 44425023df5e8cc9c51f5f004aeebfaf6286e0f4

Showing 1 changed file with 9 additions and 8 deletions.
Split Unified

@@ -1111,14 +1111,15 @@ def fecha_hoy():
1111     print("La fecha no es correcta")
1112
1113     #R11 ----- edad_hoy
1114     -----
1115     -----
1116     -----
1117     -----
1118     -----
1119     -----
1120     -----
1121     -----
1122
1123     #-----Pruebas-----
1124     #R6
1125
```

1.2.11. V5

- Se corrigieron algunas pulgas que ocurrían en las funciones `edad_al` y `edad_hoy`. Además, se agregó una función complementaria llamada `contador_dia_simple`.

1.2.12. Diferencias entre componentes

Showing 1 changed file with 79 additions and 21 deletions.

Split Unified

```
Asignación 1b - 2016000309 - 2018093728/Asignación 1b - 2016000309 - 2018093728.py
...

36 NumDia = (dia+((13*mes-1)//5)+UD*(UD//4)+(PD//4)-2*PD)*N7 #Numero
37 correspondiente al día de la semana de esa fecha en particular
38 return NumDia

36 NumDia = (dia+((13*mes-1)//5)+UD*(UD//4)+(PD//4)-2*PD)*N7 #Numero
37 correspondiente al día de la semana de esa fecha en particular
38 return NumDia

39 + def contador_dia_simple(fecha):
40 +     Dias_31 = [1,3,5,7,8,10,12]
41 +     Dias_30 = [4,6,9,11]
42 +     anno = fecha[0]
43 +     dia = fecha[2]
44 +     mes = fecha[1]
45 +     if (mes == 0 and dia == 31):
46 +         fecha = (anno,mes+1,1)
47 +         return fecha
48 +     elif (mes in Dias_31 and (dia == 31)):
49 +         if (mes == 12): #Devuelve el año siguiente
50 +             fecha = (anno+1,1,1)
51 +             return fecha
52 +     else:
53 +         fecha = (anno,mes+1,1)
54 +         return fecha
55 +
56 +     elif (mes in Dias_30 and (dia == 30)):
57 +         fecha = (anno,mes+1,1)
58 +         return fecha
59 +
60 +     elif (bisiesto(anno)) and (mes == 2):
61 +         if (dia == 29):
62 +             fecha = (anno,mes+1,1)
63 +             return fecha
64 +         else:
65 +             fecha = (anno,mes,dia+1)
66 +             return fecha
67 +
68 +     elif (mes == 2 and (dia == 28)):
69 +         fecha = (anno,mes+1,1)#Devuelve el día siguiente en Febrero
70 +         return fecha
71 +     else:
72 +         fecha = (anno,mes,dia+1)
73 +         return fecha
74 +

39 #R1 -----
40 #Joshua
41 def bisiesto(fecha): #Cambio de nombre

75 #R1 -----
76 #Joshua
77 def bisiesto(fecha): #Cambio de nombre

39 + def dias_entre(fecha1,fecha2):
40 +     print("La fecha no es válida")
41 +
1060 print("La fecha no es válida")
1061
1062 #R9 ----- edad_al
1063 -----
1064 -----
1065 -----
1066 def edad_al(fecha_na,fecha_actual):
1067     Dias_31 = [1,3,5,7,8,10,12]
1068     Dias_30 = [4,6,9,11]
1069     tiempo=(0,0,0)
1070
1071     if(fecha_es_valida(fecha_na)):
1072         if(fecha_es_valida(fecha_actual)):
1073             if (fecha_na[0]==fecha_actual[0] and fecha_na[1]==fecha_actual[1]
1074             and fecha_na[2]>fecha_actual[2]):
1075                 print("La fecha de nacimiento es mas reciente que la fecha
1076                 actual dada")
1077             else:
1078                 annos= fecha_actual[0] - fecha_na[0]
1079                 if((fecha_actual[2] - fecha_na[2])>0):
1080                     meses = fecha_actual[1] - fecha_na[1]
1081                     dias = fecha_actual[2] - fecha_na[2]
1082                 else:
1083                     meses = fecha_actual[1] - fecha_na[1] -1
1084                     if (fecha_actual[1]-1 in Dias_31 ):
1085                         dias = 31 - fecha_na[2] + fecha_actual[2]
1086                     elif (fecha_actual[1]-1 in Dias_30 ):
1087                         dias = 30 - fecha_na[2] + fecha_actual[2]
1088                     else:
1089                         if(bisiesto(fecha_actual[0])):
1090                             dias = 29 - fecha_na[2] + fecha_actual[2]
1091                         else:
1092                             dias = 28 - fecha_na[2] + fecha_actual[2]
1093                     tiempo= (annos,meses,dias)
1094
1095     else:
1096         print("La fecha nacimiento dada no es válida")
1097
1098 #R10 ----- fecha_hoy
1099 -----
1100 #Descripción: Llamar a esta función devuelve la fecha de hoy, obtenida a partir
1101 de la función today(), pero con el formato (año,mes,día)

1127 + def edad_hoy(fecha_na):
1128 +     #dia_semana((2022,4,8))
1129 +     #dia_semana((1223,4,7))
1130 +     #dia_semana((123144,24,4))

1170 + #dia_semana((2022,4,8))
1171 + #dia_semana((1223,4,7))
1172 + #dia_semana((123144,24,4))
1173 + #?
1174 + #fecha_futura((2022,2,7),2)
1175 + #fecha_futura((2022,2,28),2)
1176 + #fecha_futura((2022,2,28),0)

1092 return tiempo
1093
1094 else:
1095     print("La fecha actual dada no es válida")
1096
1097 else:
1098     print("La fecha nacimiento dada no es válida")
1099
1100 #R10 ----- fecha_hoy
1101 -----
1102 #Descripción: Llamar a esta función devuelve la fecha de hoy, obtenida a partir
1103 de la función today(), pero con el formato (año,mes,día)

1122 + while(fecha_na != fecha_actual):
1123 +     esBisiesto = bisiesto(fecha_na[0])
1124 +     anno = fecha_na[0]
1125 +     mes = fecha_na[1]
1126 +     dia = fecha_na[2]
1127 +     if (mes in Dias_31 and (dia == 31)):
1128 +         if (mes == 12): #Devuelve el año siguiente
1129 +             fecha_na = (anno+1,1,1)
1130 +             tiempo = contador_dia_simple(tiempo)
1131 +
1132 +         elif (mes == 2 and (dia == 28)):
1133 +             fecha_na = (anno,mes+1,1)#Devuelve el día siguiente en
1134 +             Febrero
1135 +             tiempo = contador_dia_simple(tiempo)
1136 +         else:
1137 +             fecha_na = (anno,mes,dia+1)
1138 +             tiempo = contador_dia_simple(tiempo)
1139 +         tiempo = (tiempo[0],tiempo[1]-1,tiempo[2])
1140 +     return tiempo
1141
1142 else:
1143     print("La fecha actual dada no es válida")
1144
1145 else:
1146     print("La fecha nacimiento dada no es válida")
1147
1148 #R10 ----- fecha_hoy
1149 -----
1150 #Descripción: Llamar a esta función devuelve la fecha de hoy, obtenida a partir
1151 de la función today(), pero con el formato (año,mes,día)
```


1.2.13. V6

- En las funciones edad_al y edad_hoy, se agregó la validación para que tomara en cuenta los años bisiestos, respecto a los días.

1.2.14. Diferencias entre componentes

Showing 1 changed file with 8 additions and 1 deletion.

Split	Unified
<pre>@@ -1124,7 ~1124,14 @@ def edad_al(fecha_na, fecha_actual): 1124 elif (mes in Dias_30) and (dia == 30): 1125 fecha_na = (anno, mes-1, 1) 1126 tiempo = contador_dia_simple(tiempo) 1127 - elif (mes == 2) and (dia == 28):</pre>	<pre>1124 elif (mes in Dias_30) and (dia == 30): 1125 fecha_na = (anno, mes-1, 1) 1126 tiempo = contador_dia_simple(tiempo) 1127 + elif (mes == 2) and (esBisiesto==bool(1)) and (dia == 29): 1128 + if (dia == 29): 1129 + fecha_na = (anno, mes+1, 1) # Devuelve 1130 + el día siguiente en Febrero bisiesto 1131 + tiempo = contador_dia_simple(tiempo) 1132 + if (dia == 28): 1133 + fecha_na = (anno, mes, dia + 1) # Devuelve 1134 + el día 29 Febrero 1135 + tiempo = contador_dia_simple(tiempo) 1136 + elif (mes == 2) and (esBisiesto==bool(0)) and (dia == 28): 1137 + fecha_na = (anno, mes+1, 1)#Devuelve el día siguiente en 1138 + Febrero 1139 + tiempo = contador_dia_simple(tiempo) 1140 + else:</pre>

2. Asignacion 1b pruebas

2.1. R6 (dia_semana)

Dada una fecha válida, determinar el día de la semana que le corresponde, con la siguiente codificación: 0 = domingo, 1 = lunes, 2 = martes, 3 = miércoles, 4 = jueves, 5 = viernes, 6 = sábado. El resultado debe ser un número entero, conforme a la codificación indicada

```
>>> dia_semana((2022,4,7))
4
>>> dia_semana((2022,4,8))
5
>>> dia_semana((1223,4,7))
Fecha no valida
>>> dia_semana((123144,24,4))
Fecha no valida
>>> dia_semana((2022,4,72))
Fecha no valida
>>> dia_semana((2022,40,7))
Fecha no valida
>>> dia_semana((2022,4,9))
6
>>> dia_semana((2022,4,10))
0
>>> dia_semana((2022,4,11))
1
>>> dia_semana((2022,4,12))
2
>>> dia_semana((2022,4,13))
3
```

2.2. R7 (fecha_futura)

Dados una fecha válida f y un número entero no-negativo n , determinar la fecha que está n días naturales en el futuro. Si n es 0, entonces $\text{fecha_futura}(f, 0) = f$. El resultado debe ser una fecha válida

```

>>> fecha_futura((2022,2,7),2)
      (2022, 2, 9)
>>> fecha_futura((2022,2,28),2)
      (2022, 3, 2)
>>> fecha_futura((2022,2,28),0)
      (2022, 2, 28)
>>> fecha_futura((2022,2,29),0)
La fecha no es valida
>>> fecha_futura((2022,2,29),1)
La fecha no es valida
>>> fecha_futura((2016,2,29),1)
      (2016, 3, 1)
>>> fecha_futura((2016,13,2),1)
La fecha no es valida
>>> fecha_futura((1333,13,2),1)
La fecha no es valida
>>> fecha_futura((1993,12,31),1)
      (1994, 1, 1)
>>> fecha_futura((1993,12,31),20)
      (1994, 1, 20)
>>> fecha_futura((1993,12,31),365)
      (1995, 1, 8)
>>> fecha_futura((1993,12,31),-365)
La numero es negativo

```

2.3. R8 (dias_entre)

Dadas dos fechas válidas, $f1$ y $f2$, sin importar si $f1 < f2$ o $f2 < f1$, determinar el número de días naturales entre las dos fechas. Si $f1 = f2$, entonces $\text{dias_entre}(f1, f2) = 0$. El resultado debe ser un número entero no negativo.

```

>>> dias_entre((2022,4,7),(2022,4,6))
1
>>> dias_entre((2022,4,6),(2022,4,7))
1
>>> dias_entre((2022,4,7),(2022,4,7))
0
>>> dias_entre((2022,5,7),(2022,4,6))
31
>>> dias_entre((2022,4,7),(2022,5,6))
29
>>> dias_entre((2023,4,7),(2022,4,6))
366
>>> dias_entre((2022,4,7),(2023,4,6))
364
>>> dias_entre((2022,4,7),(2022,4,6))
1
>>> dias_entre((1333,4,7),(2022,4,6))
La fecha no es válida
>>> dias_entre((1583,4,7),(2022,4,6))
160344
>>> dias_entre((2022,4,7),(2022,53,6))
La fecha no es válida
>>> dias_entre((2022,4,7),(2022,53,6))
...
La fecha no es válida
>>> dias_entre((2022,64,7),(2022,4,6))
La fecha no es válida
>>> dias_entre((2022,4,7),(2022,4,51))
La fecha no es válida
>>> dias_entre((2022,4,91),(2022,4,6))
La fecha no es válida

```

2.4. R9 (edad_al)

Dadas dos fechas válidas, f1 y f2, donde f1 representa una fecha de nacimiento y f2 es tal que f1 ≤ f2, determinar la edad de la persona en años, meses y días cumplidos desde la fecha f1 hasta la fecha f2. El resultado debe ser una tupla (año, mes, día); note que – en este caso – tal tupla no es necesariamente una fecha válida, sino una tupla con los tres componentes enteros requeridos. El resultado debe ser una tupla de tres números enteros no negativos.

```
>>> edad_al((1998,23,20),(2022,4,8))
La fecha nacimiento dada no es válida
>>> edad_al((1998,3,20),(2022,14,8))
La fecha actual dada no es válida
>>> edad_al((2000,2,28),(2000,3,1))
(0, 0, 2)
>>> edad_al((2001,2,28),(2001,3,1))
(0, 0, 1)
>>> edad_al((1998,3,20),(2022,4,8))
(24, 0, 25)
>>> edad_al((1998,12,20),(2022,2,29))
La fecha actual dada no es válida
>>> edad_al((1998,12,20),(2022,2,22))
(23, 2, 11)
```

2.5. R10 (fecha_hoy)

Obtener la fecha ‘del sistema’ a partir de la invocación de la función de biblioteca de Python today() , luego convertirla a una fecha válida – tupla (año, mes, día) – en el calendario gregoriano, según las convenciones usadas en esta asignación y la anterior. El resultado debe ser una fecha válida.

```

>>> edad_hoy((1998,3,20))
(24, 0, 25)
>>> edad_hoy((1998,31,20))
La fecha nacimiento dada no es válida
>>> edad_hoy((1998,2,28))
(24, 1, 14)
>>> edad_hoy((1998,2,29))
La fecha nacimiento dada no es válida
>>> edad_hoy((2000,2,29))
(22, 1, 13)
>>> edad_hoy((1853,1,29))
(169, 3, 21)
>>> edad_hoy((1582,1,29))
La fecha nacimiento dada no es válida
>>> edad_hoy((2000,10,18))
(21, 5, 26)

```

2.6. R11 (edad_hoy):

Dada una fecha válida f , que corresponde a la fecha de nacimiento de alguna persona, determinar la edad de la persona en años, meses y días cumplidos desde la fecha f hasta la fecha de hoy. Suponemos que f es menor o igual que la fecha de hoy. El resultado debe ser una tupla (año, mes, día); note que – en este caso – eso no es una fecha válida, sino una tupla con los tres componentes requeridos. El resultado debe ser una tupla de tres números enteros no negativos.

```

>>> edad_hoy((1998,3,20))
(24, 0, 19)
>>> edad_hoy((1998,31,20))
La fecha nacimiento dada no es válida
>>> edad_hoy((1998,31,1643))
La fecha nacimiento dada no es válida
>>> edad_hoy((1333,2,1))
La fecha nacimiento dada no es válida
>>> edad_hoy((1965,12,31))
(56, 3, 8)

```

3. Desglose de tiempo

3.1. Mariana

Actividad Realizada	Horas
Investigación acerca del dominio del problema	0.25
Diseño de la solución	0.25
Programación	2.50
Pruebas	1.00
Correcciones y re-trabajo	1.75
Documentación	1.50
Integración final de la entrega	1.00

3.2. Joshua

Actividad Realizada	Horas
Investigación acerca del dominio del problema	1.00
Diseño de la solución	0.50
Programación	3.00
Pruebas	2.00
Correcciones y re-trabajo	3.00
Documentación	1.50
Integración final de la entrega	1.00

3.3. Joshua y Mariana

Actividad Realizada	Horas
Investigación acerca del dominio del problema	1.25
Diseño de la solución	0.75
Programación	5.50
Pruebas	3.00
Correcciones y re-trabajo	4.75
Documentación	3.00
Integración final de la entrega	2.00

4. Código

4.1. R6 (dia_semana)

```
1 def dia_semana(fecha):
2     if (fecha_es_tupla(fecha) and fecha_es_valida(fecha) and
3         fecha[0] > 1582 ): #Se agregaron validaciones
4         formatoMeses =
5             {1:11,2:12,3:1,4:2,5:3,6:4,7:5,8:6,9:7,10:8,11:9,12:10} #
6             Valor n merico que toman los meses en este algoritmo
7             dia = fecha[2] #Dia del a o
8             mes = formatoMeses[fecha[1]] #Mes del a o , se cuenta
9             desde marzo (Marzo = 1)
10            anno = fecha[0]
11            if (mes == 12) or (mes == 11): #Se cuentan los ultimos
12            dos meses del a o pasado como enero y febrero
13            anno -=1
14            PD = anno//100 #Primeros dos digitos del a o
15            UD = anno%100 #Ultimos dos digitos del a o
16
17            NumDia = (dia+( (13*mes-1)//5 )+UD+(UD//4)+(PD//4)-2*PD)
18            %7 #Numero correspondiente al d a de la semana de esa fecha
19            en particular
20            return NumDia
21        else:
22            print("Fecha no valida")
```

4.2. R7 (fecha_futura)

```
1 def fecha_futura(fecha,num):
2     Dias_31 = [1,3,5,7,8,10,12]
3     Dias_30 = [4,6,9,11]
4
5     if(fecha_es_valida(fecha)):
6         if (num < 0):
7             print("La numero es negativo")
8         elif(num == 0):
9             return fecha
10        else:
11            while num>0:
12                dias_mas=0
13                if(fecha[1] in Dias_31):
14                    dias_mas = 31 - fecha[2]
15                    if (num >= dias_mas):
16                        fecha = (fecha[0],fecha[1],fecha[2]+
17                                dias_mas)
18                        num = num - dias_mas
19                    else:
20                        fecha = (fecha[0],fecha[1],fecha[2]+num)
21                        num = 0
22
23                if (fecha[2]==31):
24                    if(num >0):
25                        if (fecha[1]==12):
26                            fecha = (fecha[0]+1,1,1)
```



```

26         else:
27             fecha = (fecha[0], fecha[1]+1, 1)
28             num = num - 1
29
30         elif(fecha[1] in Dias_31):
31             dias_mas = 30 - fecha[2]
32             if (num >= dias_mas ):
33                 fecha = (fecha[0], fecha[1], fecha[2]+
34                     dias_mas)
35                 num = num - dias_mas
36             else:
37                 fecha = (fecha[0], fecha[1], fecha[2]+num)
38                 num = 0
39
40             if (fecha[2]==30):
41                 if(num >0):
42                     fecha = (fecha[0], fecha[1]+1, 1)
43                     num = num - 1
44             else:
45                 if (bisiesto(fecha[0])):
46                     dias_mas = 29 - fecha[2]
47                     if (num >= dias_mas ):
48                         fecha = (fecha[0], fecha[1], fecha[2]+
49                             dias_mas)
50                         num = num - dias_mas
51                     else:
52                         fecha = (fecha[0], fecha[1], fecha[2]+
53                             num)
54                         num = 0
55
56                 if (fecha[2]==29):
57                     if(num >0):
58                         fecha = (fecha[0], fecha[1]+1, 1)
59                         num = num - 1
60                 else:
61                     dias_mas = 28 - fecha[2]
62                     if (num >= dias_mas ):
63                         fecha = (fecha[0], fecha[1], fecha[2]+
64                             dias_mas)
65                         num = num - dias_mas
66                     else:
67                         fecha = (fecha[0], fecha[1], fecha[2]+
68                             num)
69                         num = 0
70
71                 if (fecha[2]==28):
72                     if(num >0):
73                         fecha = (fecha[0], fecha[1]+1, 1)
74                         num = num - 1
75
76         return fecha
77     else:
78         print("La fecha no es valida")

```

4.3. R8 (dias_entre)

```

1 def dias_entre(fecha1, fecha2):

```

```

2  if (fecha_es_tupla(fecha1) and fecha_es_tupla(fecha2) and
    fecha_es_valida(fecha1) and fecha_es_valida(fecha2) and
    fecha1[0] > 1582 and fecha2[0] > 1582): #Validaci n con R0 y
    R2
3      if (fecha1 == fecha2):
4          return 0
5      elif (fecha1[0] == fecha2[0]):
6          if (fecha1[1] == fecha2[1]):
7              if (fecha1[2] > fecha2[2]):
8                  return fecha1[2] - fecha2[2]
9              else:
10                 return fecha2[2] - fecha1[2]
11         else:
12             if (fecha1[1] > fecha2[1]):
13                 return ordinal_dia(fecha1) - ordinal_dia(
    fecha2)
14             else:
15                 return ordinal_dia(fecha2) - ordinal_dia(
    fecha1)
16         else:
17             dias = 0
18             if (fecha1[0] > fecha2[0]):
19                 while (fecha1 > fecha2):
20                     fecha2 = dia_siguiente(fecha2)
21                     dias += 1
22             return dias
23         else:
24             while (fecha2 > fecha1):
25                 fecha1 = dia_siguiente(fecha1)
26                 dias += 1
27             return dias
28     else:
29         print("La fecha no es v lida")

```

4.4. R9 (edad_al)

```

1  def edad_al(fecha_na, fecha_actual):
2      Dias_31 = [1,3,5,7,8,10,12]
3      Dias_30 = [4,6,9,11]
4      tiempo=(0,1,0)
5
6      if(fecha_es_valida(fecha_na)):
7          if (fecha_es_valida(fecha_actual)):
8              if (fecha_na[0]==fecha_actual[0] and fecha_na[1]>=
    fecha_actual[1] and fecha_na[2]>fecha_actual[2]):
9                  print("La fecha de nacimiento es mas reciente que
    la fecha actual dada")
10             else:
11                 while(fecha_na != fecha_actual):
12                     esBisiesto = bisiesto(fecha_na[0])
13                     anno = fecha_na[0]
14                     mes = fecha_na[1]
15                     dia = fecha_na[2]
16                     if (mes in Dias_31) and (dia == 31):
17                         if (mes == 12): #Devuelve el a o
    siguiente

```

```

18         fecha_na = (anno+1,1,1)
19         tiempo = contador_dia_simple(tiempo)
20     else:
21         fecha_na = (anno,mes+1,1)
22         tiempo = contador_dia_simple(tiempo)
23     elif (mes in Dias_30) and (dia == 30):
24         fecha_na = (anno,mes+1,1)
25         tiempo = contador_dia_simple(tiempo)
26     elif (mes == 2) and (esBisiesto==bool(1)) and
(dia == 29):
27         if (dia == 29):
28             fecha_na = (anno,mes+1,1)
29             # Devuelve el d a siguiente en Febrero bisiesto
30             tiempo = contador_dia_simple(tiempo)
31             if (dia == 28):
32                 fecha_na = (anno,mes,dia + 1)
33                 # Devuelve el d a 29 Febrero
34                 tiempo = contador_dia_simple(tiempo)
35             elif (mes == 2) and (esBisiesto==bool(0)) and
(dia == 28):
36                 fecha_na = (anno,mes+1,1)#Devuelve el
37                 d a siguiente en Febrero
38                 tiempo = contador_dia_simple(tiempo)
39             else:
40                 fecha_na = (anno,mes,dia+1)
41                 tiempo = contador_dia_simple(tiempo)
42                 tiempo = (tiempo[0],tiempo[1]-1,tiempo[2])
43                 return tiempo
44     else:
45         print("La fecha actual dada no es v lida")
46 else:
47     print("La fecha nacimiento dada no es v lida")

```

4.5. R10 (fecha_hoy)

```

1 def fecha_hoy():
2     fecha_sin_formatear = str(date.today()) # Obtiene la fecha de
3     hoy
4     anno = int(fecha_sin_formatear[0:4])
5     mes = int(fecha_sin_formatear[5:7])
6     dia = int(fecha_sin_formatear[8:10])
7     fecha_formateada = (anno,mes,dia)
8     if (fecha_es_tupla(fecha_formateada) and fecha_es_valida(
9     fecha_formateada) and fecha_formateada[0] > 1582 ):
10         return fecha_formateada
11     else:
12         print("La fecha no es correcta")

```

4.6. R11 (edad_hoy):

```

1 def edad_hoy(fecha_na):
2     tiempo=(0,0,0)
3
4     if(fecha_es_valida(fecha_na)):

```

```

5     fecha_actual = fecha_hoy()
6     tiempo = edad_al(fecha_na, fecha_actual)
7     return tiempo
8 else:
9     print("La fecha nacimiento dada no es v lida")

```

4.7. Código base

Acontinucion se muestra el codigo de la asignacion 1a con los cambios realizados para utilizarlos en la asignacion 1b.

```

1  #-----Referencias
2  #https://pyformat.info/#string_pad_align
3  #https://docs.python.org/3/library/datetime.html
4
5  from datetime import date
6  #RO
7
8  #mari
9  def fecha_es_tupla(fecha): #Cambio de nombre
10     if(len(fecha) != 3):
11         #print('tupla no valida')
12         return (bool(0))
13     else:
14         for i in range(3):
15             if (fecha[i] <= 0):
16                 #print('no todos son enteros positivos')
17                 return (bool(0))
18             if (fecha[1] <= 12):
19                 if (fecha[2] <= 32):
20                     return (bool(1))
21             else:
22                 #print('Mes no valida')
23                 return (bool(0))
24 # (15,1,12) = valido      (2000,2,0)=falso
25
26 def cal_Dia(fecha):
27     formatoMeses =
28     {1:11,2:12,3:1,4:2,5:3,6:4,7:5,8:6,9:7,10:8,11:9,12:10}
29     dia = fecha[2] #Dia del a o
30     mes = formatoMeses[fecha[1]] #Mes del a o , se cuenta desde
31     marzo (Marzo = 1)
32     anno = fecha[0]
33     if (mes == 12) or (mes == 11): #Se cuentan los ultimos dos
34     meses del a o pasado como enero y febrero
35         anno -=1
36     PD = anno//100 #Primeros dos digitos del a o
37     UD = anno%100 #Ultimos dos digitos del a o
38
39     NumDia = (dia+( (13*mes-1)//5 )+UD+(UD//4)+(PD//4)-2*PD)%7 #
40     Numero correspondiente al d a de la semana de esa fecha en
41     particular
42     return NumDia

```

```

38
39 def contador_dia_simple(fecha):
40     Dias_31 = [1,3,5,7,8,10,12]
41     Dias_30 = [4,6,9,11]
42     anno = fecha[0]
43     dia = fecha[2]
44     mes = fecha[1]
45     if (mes == 0 and dia == 31):
46         fecha = (anno,mes+1,1)
47         return fecha
48     elif (mes in Dias_31) and (dia == 31):
49         if (mes == 12): #Devuelve el a o siguiente
50             fecha = (anno+1,1,1)
51             return fecha
52         else:
53             fecha = (anno,mes+1,1)
54             return fecha
55
56     elif (mes in Dias_30) and (dia == 30):
57         fecha = (anno,mes+1,1)
58         return fecha
59
60     elif (bisiesto(anno)) and (mes == 2):
61         if (dia == 29):
62             fecha = (anno,mes+1,1)
63             return fecha
64         else:
65             fecha = (anno,mes,dia+1)
66             return fecha
67
68     elif (mes == 2) and (dia == 28):
69         fecha = (anno,mes+1,1)#Devuelve el d a siguiente en
70         Febrero
71         return fecha
72     else:
73         fecha = (anno,mes,dia+1)
74         return fecha
75 #R1
76 -----
77
78 #joshua
79 def bisiesto(fecha): #Cambio de nombre
80     if(fecha >= 1582):
81         if(fecha%4 == 0):
82             return True
83         elif (fecha%100 == 0) and (fecha%400 == 0):
84             return True
85         else:
86             return False
87
88 #R2 ----- fecha valida G.
89 -----
90
91 #mari
92
93 def cal_G(dia, mes, anno): #Cambio de nombre
94     Dias_31 = [1,3,5,7,8,10,12]

```

```

91 Dias_30 = [4,6,9,11]
92
93 if(anno==1582):
94     if (mes==10 and dia>=15 and dia<=31):
95         return (bool(1))
96     elif (mes==11 and dia<=30):
97         return (bool(1))
98     elif (mes==12 and dia<=31):
99         return (bool(1))
100     else:
101         return (bool(0))
102
103 elif(mes==2 and dia<=28):
104     return (bool(1))
105
106 elif (bisiesto(anno)and dia<=29):
107     return (bool(1))
108
109 elif(mes in Dias_30 and dia<=30):
110     #print(dia)
111     return (bool(1))
112
113 elif(mes in Dias_31 and dia<=31):
114     return (bool(1))
115
116 else:
117     return (bool(0))
118
119 def fecha_es_valida(fecha):
120     if(fecha_es_tupla(fecha)):
121         if(fecha[0]>=1582):
122             if (cal_G(fecha[2], fecha[1], fecha[0])):
123                 return (bool(1))
124             else:
125                 return (bool(0))
126         else:
127             return (bool(0))
128     else:
129         return (bool(0))
130
131
132 #R3 ----- dia siguiente
133 #joshua
134 def dia_siguiente(dia_actual): #Cambio de nombre
135     Dias_31 = [1,3,5,7,8,10,12]
136     Dias_30 = [4,6,9,11]
137     if (fecha_es_tupla(dia_actual) and fecha_es_valida(dia_actual)
138         ) and dia_actual[0] > 1582): #Normalizaci n de espacios e
139         identaciones
140         esBisiesto = bisiesto(dia_actual[0])
141         dia = dia_actual[2]
142         mes = dia_actual[1]
143         if (mes in Dias_31) and (dia == 31):
144             if (mes == 12): #Devuelve el a o siguiente
145                 dia_actual = (dia_actual[0]+1,1,1)
146             return dia_actual

```

```

145         else:
146             dia_actual = (dia_actual[0], dia_actual[1]+1, 1)
147             return dia_actual
148         elif (mes in Dias_30) and (dia == 30):
149             dia_actual = (dia_actual[0], dia_actual[1]+1, 1)
150             return dia_actual
151
152         elif (esBisiesto) and (mes == 2):
153             if (dia == 29):
154                 dia_actual = (dia_actual[0], dia_actual[1]+1, 1)
155                 return dia_actual
156             else:
157                 dia_actual = (dia_actual[0], dia_actual[1],
158                             dia_actual[2]+1)
159                 return dia_actual
160             elif (mes == 2) and (dia == 28):
161                 dia_actual = (dia_actual[0], dia_actual[1]+1, 1) #
162                 Devuelve el d a siguiente en Febrero
163                 return dia_actual
164             else:
165                 dia_actual = (dia_actual[0], dia_actual[1], dia_actual
166                             [2]+1)
167                 return dia_actual
168         else:
169             print("La fecha ingresada no es v lida")
170
171 #R4 ----- ordinal
172 -----
173 #mari
174 def ordinal_dia(ordinal): #Cambio de nombre
175     Dias_31 = [1,3,5,7,8,10,12]
176     Dias_30 = [4,6,9,11]
177     dias = 0
178     if (fecha_es_valida(ordinal)):
179         for i in range(ordinal[1]):
180             if (i == 2):
181                 if (bisiesto(ordinal[0])):
182                     #print("isiesto")
183                     dias = dias + 29
184                 else:
185                     dias = dias + 28
186             elif (i in Dias_30):
187                 dias = dias + 30
188             elif (i in Dias_31):
189                 dias = dias + 31
190             dias = dias + ordinal[2]
191         return (dias)
192
193 #R5 ----- CALENDARIO
194 -----
195 #joshua y mari
196 def Imprimir_3x4(calendario): #Cambio de nombre
197     Dias_31 = [1,3,5,7,8,10,12]
198     Dias_30 = [4,6,9,11]
199     if calendario <= 1582:
200         print("La fecha no es correcta")

```

```

197     else:
198         Meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio",
199                 "Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "
200                 Diciembre"]
201         Dias = "    D  L  M  M  J  V  S  "
202         x = 1
203         print("
204
205 -----
206
207         print("                                Calendario
208         ", calendario)
209         while (x < 4):
210             #print(' {:^10}{:^10}{:^10}{:^10} '.format("Enero", "
211             Febrero", "Marzo", "Abril"))
212             if x == 1:
213                 print("
214
215 -----
216
217                 print(' {:^15}{:^45}{:^5}{:^45} '.format("Enero",
218                 "Febrero", "Marzo", "Abril"))
219                 print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
220                 format("D", "L", "K", "M", "J", "V", "S")*3, end = "")
221                 print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
222                 format("D", "L", "K", "M", "J", "V", "S"))
223                 m1 = (calendario, 1, 1)
224                 d1 = (calendario, 1, 1)
225                 d2 = (calendario, 2, 1)
226                 d3 = (calendario, 3, 1)
227                 d4 = (calendario, 4, 1)
228                 elif x == 2:
229                     print("
230
231 -----
232
233                     print(' {:^15}{:^45}{:^5}{:^45} '.format("Mayo", "
234                     Junio", "Julio", "Agosto"))
235                     print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
236                     format("D", "L", "K", "M", "J", "V", "S")*3, end = "")
237                     print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
238                     format("D", "L", "K", "M", "J", "V", "S"))
239                     m1 = (calendario, 5, 1)
240                     d1 = (calendario, 5, 1)
241                     d2 = (calendario, 6, 1)
242                     d3 = (calendario, 7, 1)
243                     d4 = (calendario, 8, 1)
244
245                 else:
246                     print("
247
248 -----
249
250                     print(' {:^20}{:^35}{:^5}{:^45} '.format("
251                     Septiembre", "Octubre", "Noviembre", "Diciembre"))
252                     print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
253                     format("D", "L", "K", "M", "J", "V", "S")*3, end = "")
254                     print('| {:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}{:^3}| '.
255                     format("D", "L", "K", "M", "J", "V", "S"))
256                     m1 = (calendario, 9, 1)
257                     d1 = (calendario, 9, 1)
258                     d2 = (calendario, 10, 1)
259                     d3 = (calendario, 11, 1)

```



```

235         d4 = (calendario,12,1)
236
237         c = 0
238         i=0
239         j = 0
240         f1=0
241         f2=0
242         f3=0
243         f4=0
244         while (i < 24):
245             p1 = " "
246             p2 = " "
247             p3 = " "
248             p4 = " "
249             p5 = " "
250             p6 = " "
251             p7 = " "
252             if (i == 0 or i == 4 or i == 8 or i == 12 or i ==
16 or i == 20):
253                 m1 = d1
254                 ff = f1
255                 if (i > 0):
256                     m1 = R3(m1)
257             elif (i == 1 or i == 5 or i == 9 or i == 13 or i
== 17 or i == 21):
258                 m1 = d2
259                 ff = f2
260                 if (i > 1):
261                     m1 = R3(m1)
262             elif (i == 2 or i == 6 or i == 10 or i == 14 or i
== 18 or i == 22):
263                 m1 = d3
264                 ff = f3
265                 if (i > 2):
266                     m1 = R3(m1)
267             elif (i == 3 or i == 7 or i == 11 or i == 15 or i
== 19 or i == 23):
268                 m1 = d4
269                 ff = f4
270                 if (i > 3):
271                     m1 = R3(m1)
272             c = cal_Dia(m1)
273             if (c == 0 ):
274                 if(m1[1]== 2):
275                     if(bisiesto(m1[0])):
276                         if (m1[2]<=29 and ff==0):
277                             if (m1[2]==29):
278                                 ff = 1
279                             p1 = str(m1[2])
280
281                     m1 = R3(m1)
282                     if (m1[2]<=29 and ff==0):
283                         if (m1[2]==29):
284                             ff = 1
285                         p2 = str(m1[2])
286
287                     m1 = R3(m1)

```

```

288         if (m1[2]<=29 and ff==0):
289             if (m1[2]==29):
290                 ff = 1
291                 p3 = str(m1[2])
292
293             m1 = R3(m1)
294             if (m1[2]<=29 and ff==0):
295                 if (m1[2]==29):
296                     ff = 1
297                     p4 = str(m1[2])
298
299             m1 = R3(m1)
300             if (m1[2]<=29 and ff
==0):
301                 if (m1[2]==29):
302                     ff = 1
303                     p5 = str(m1[2])
304
305                 m1 = R3(m1)
306                 if (m1[2]<=29 and
ff==0):
307                     if (m1
[2]==29):
308                         ff = 1
309                         p6 = str(m1
[2])
310
311                     m1 = R3(m1)
312                     if (m1[2]<=29
and ff==0):
313                         if (m1
[2]==29):
314                             ff =
1
315                             p7 = str(
m1[2])
316
317             else:
318                 if (m1[2]<=28 and ff==0):
319                     if (m1[2]==28):
320                         ff = 1
321                         p1 = str(m1[2])
322
323                     m1 = R3(m1)
324                     if (m1[2]<=28 and ff==0):
325                         if (m1[2]==28):
326                             ff = 1
327                             p2 = str(m1[2])
328
329                     m1 = R3(m1)
330                     if (m1[2]<=28 and ff==0):
331                         if (m1[2]==28):
332                             ff = 1
333                             p3 = str(m1[2])
334
335                     m1 = R3(m1)
336                     if (m1[2]<=28 and ff==0):
337                         if (m1[2]==28):

```

```

337         ff = 1
338         p4 = str(m1[2])
339
340         m1 = R3(m1)
341         if (m1[2]<=28 and ff
==0):
342             if (m1[2]==28):
343                 ff = 1
344                 p5 = str(m1[2])
345
346             m1 = R3(m1)
347             if (m1[2]<=28 and
ff==0):
348                 if (m1
[2]==28):
349                     ff = 1
350                     p6 = str(m1
[2])
351
352                     m1 = R3(m1)
353                     if (m1[2]<=28
and ff==0):
354                         if (m1
[2]==28):
355                             ff =
1
356                             p7 = str(
m1[2])
357
358         if(m1[1] in Dias_30):
359             if (m1[2]<=30 and ff==0):
360                 if (m1[2]==30):
361                     ff = 1
362                     p1 = str(m1[2])
363
364             m1 = R3(m1)
365             if (m1[2]<=30 and ff==0):
366                 if (m1[2]==30):
367                     ff = 1
368                     p2 = str(m1[2])
369
370             m1 = R3(m1)
371             if (m1[2]<=30 and ff==0):
372                 if (m1[2]==30):
373                     ff = 1
374                     p3 = str(m1[2])
375
376             m1 = R3(m1)
377             if (m1[2]<=30 and ff==0):
378                 if (m1[2]==30):
379                     ff = 1
380                     p4 = str(m1[2])
381
382             m1 = R3(m1)
383             if (m1[2]<=30 and ff==0):
384                 if (m1[2]==30):
385                     ff = 1
386                     p5 = str(m1[2])

```

```

386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435

==0):
    m1 = R3(m1)
    if (m1[2]<=30 and ff

    if (m1[2]==30):
        ff = 1
        p6 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=30 and

ff==0):
    if (m1

[2]==30):
        ff = 1
        p7 = str(m1

[2])

    if(m1[1] in Dias_31):
        if (m1[2]<=31 and ff==0):
            if (m1[2]==31):
                ff = 1
            p1 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
        p2 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
        p3 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
        p4 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
        p5 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and ff

==0):
        if (m1[2]==31):
            ff = 1
        p6 = str(m1[2])

    m1 = R3(m1)
    if (m1[2]<=31 and

ff==0):

```

```

436                                     if (m1
[2]==31):
437                                     ff = 1
438                                     p7 = str(m1
[2])
439             elif (c == 1):
440                 if (m1[1]== 2):
441                     if(bisiesto(m1[0])):
442                         if (m1[2]<=29 and ff==0):
443                             if (m1[2]==29):
444                                 ff = 1
445                                 p2 = str(m1[2])
446
447                                 m1 = R3(m1)
448                                 if (m1[2]<=29 and ff==0):
449                                     if (m1[2]==29):
450                                         ff = 1
451                                         p3 = str(m1[2])
452
453                                 m1 = R3(m1)
454                                 if (m1[2]<=29 and ff==0):
455                                     if (m1[2]==29):
456                                         ff = 1
457                                         p4 = str(m1[2])
458
459                                 m1 = R3(m1)
460                                 if (m1[2]<=29 and ff==0):
461                                     if (m1[2]==29):
462                                         ff = 1
463                                         p5 = str(m1[2])
464
465                                 m1 = R3(m1)
466                                 if (m1[2]<=29 and ff
==0):
467                                     if (m1[2]==29):
468                                         ff = 1
469                                         p6 = str(m1[2])
470
471                                     m1 = R3(m1)
472                                     if (m1[2]<=29 and
ff==0):
473                                         if (m1
[2]==29):
474                                             ff = 1
475                                             p7 = str(m1
[2])
476                                     else:
477                                         if (m1[2]<=28 and ff==0):
478                                             if (m1[2]==28):
479                                                 ff = 1
480                                                 p2 = str(m1[2])
481
482                                         m1 = R3(m1)
483                                         if (m1[2]<=28 and ff==0):
484                                             if (m1[2]==28):
485                                                 ff = 1
486                                                 p3 = str(m1[2])

```

```

487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539

m1 = R3(m1)
if (m1[2]<=28 and ff==0):
    if (m1[2]==28):
        ff = 1
    p4 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=28 and ff==0):
    if (m1[2]==28):
        ff = 1
    p5 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=28 and ff

==0):

    if (m1[2]==28):
        ff = 1
    p6 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=28 and

ff==0):

    if (m1

[2]==28):

        ff = 1
    p7 = str(m1

[2])

if(m1[1] in Dias_30):
    if (m1[2]<=30 and ff==0):
        if (m1[2]==30):
            ff = 1
        p2 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=30 and ff==0):
    if (m1[2]==30):
        ff = 1
    p3 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=30 and ff==0):
    if (m1[2]==30):
        ff = 1
    p4 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=30 and ff==0):
    if (m1[2]==30):
        ff = 1
    p5 = str(m1[2])

m1 = R3(m1)
if (m1[2]<=30 and ff==0):
    if (m1[2]==30):
        ff = 1
    p6 = str(m1[2])

```

```

540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594

==0):
    m1 = R3(m1)
    if (m1[2]<=30 and ff
        if (m1[2]==30):
            ff = 1
            p7 = str(m1[2])
    if(m1[1] in Dias_31):
        if (m1[2]<=31 and ff==0):
            if (m1[2]==31):
                ff = 1
                p2 = str(m1[2])
    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
            p3 = str(m1[2])
    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
            p4 = str(m1[2])
    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
            p5 = str(m1[2])
    m1 = R3(m1)
    if (m1[2]<=31 and ff==0):
        if (m1[2]==31):
            ff = 1
            p6 = str(m1[2])
    m1 = R3(m1)
    if (m1[2]<=31 and ff
        if (m1[2]==31):
            ff = 1
            p7 = str(m1[2])
    ==0):
        if (m1[2]==31):
            ff = 1
            p7 = str(m1[2])
    elif (c == 2):
        if(m1[1]== 2):
            if(bisiesto(m1[0])):
                if (m1[2]<=29 and ff==0):
                    if (m1[2]==29):
                        ff = 1
                        p3 = str(m1[2])
            m1 = R3(m1)
            if (m1[2]<=29 and ff==0):
                if (m1[2]==29):
                    ff = 1

```

```

595         p4 = str(m1[2])
596
597         m1 = R3(m1)
598         if (m1[2]<=29 and ff==0):
599             if (m1[2]==29):
600                 ff = 1
601                 p5 = str(m1[2])
602
603         m1 = R3(m1)
604         if (m1[2]<=29 and ff==0):
605             if (m1[2]==29):
606                 ff = 1
607                 p6 = str(m1[2])
608
609         m1 = R3(m1)
610         if (m1[2]<=29 and ff
==0):
611             if (m1[2]==29):
612                 ff = 1
613                 p7 = str(m1[2])
614
615         else:
616             if (m1[2]<=28 and ff==0):
617                 if (m1[2]==28):
618                     ff = 1
619                     p3 = str(m1[2])
620
621             m1 = R3(m1)
622             if (m1[2]<=28 and ff==0):
623                 if (m1[2]==28):
624                     ff = 1
625                     p4 = str(m1[2])
626
627             m1 = R3(m1)
628             if (m1[2]<=28 and ff==0):
629                 if (m1[2]==28):
630                     ff = 1
631                     p5 = str(m1[2])
632
633             m1 = R3(m1)
634             if (m1[2]<=28 and ff==0):
635                 if (m1[2]==28):
636                     ff = 1
637                     p6 = str(m1[2])
638
639             m1 = R3(m1)
640             if (m1[2]<=28 and ff
==0):
641                 if (m1[2]==28):
642                     ff = 1
643                     p7 = str(m1[2])
644
645         if(m1[1] in Dias_30):
646             if (m1[2]<=30 and ff==0):
647                 if (m1[2]==30):
648                     ff = 1
649                     p3 = str(m1[2])
650
651             m1 = R3(m1)

```



```

650         if (m1[2]<=30 and ff==0):
651             if (m1[2]==30):
652                 ff = 1
653                 p4 = str(m1[2])
654
655             m1 = R3(m1)
656             if (m1[2]<=30 and ff==0):
657                 if (m1[2]==30):
658                     ff = 1
659                     p5 = str(m1[2])
660
661             m1 = R3(m1)
662             if (m1[2]<=30 and ff==0):
663                 if (m1[2]==30):
664                     ff = 1
665                     p6 = str(m1[2])
666
667             m1 = R3(m1)
668             if (m1[2]<=30 and ff==0):
669                 if (m1[2]==30):
670                     ff = 1
671                     p7 = str(m1[2])
672         if(m1[1] in Dias_31):
673             if (m1[2]<=31 and ff==0):
674                 if (m1[2]==31):
675                     ff = 1
676                 p3 = str(m1[2])
677
678             m1 = R3(m1)
679             if (m1[2]<=31 and ff==0):
680                 if (m1[2]==31):
681                     ff = 1
682                 p4 = str(m1[2])
683
684             m1 = R3(m1)
685             if (m1[2]<=31 and ff==0):
686                 if (m1[2]==31):
687                     ff = 1
688                 p5 = str(m1[2])
689
690             m1 = R3(m1)
691             if (m1[2]<=31 and ff==0):
692                 if (m1[2]==31):
693                     ff = 1
694                 p6 = str(m1[2])
695
696             m1 = R3(m1)
697             if (m1[2]<=31 and ff==0):
698                 if (m1[2]==31):
699                     ff = 1
700                 p7 = str(m1[2])
701     elif (c == 3):
702         if(m1[1]== 2):
703             if(bisiesto(m1[0])):
704                 if (m1[2]<=29 and ff==0):
705                     if (m1[2]==29):
706                         ff = 1

```

```

707         p4 = str(m1[2])
708
709         m1 = R3(m1)
710         if (m1[2]<=29 and ff==0):
711             if (m1[2]==29):
712                 ff = 1
713             p5 = str(m1[2])
714
715         m1 = R3(m1)
716         if (m1[2]<=29 and ff==0):
717             if (m1[2]==29):
718                 ff = 1
719             p6 = str(m1[2])
720
721         m1 = R3(m1)
722         if (m1[2]<=29 and ff==0):
723             if (m1[2]==29):
724                 ff = 1
725             p7 = str(m1[2])
726     else:
727         if (m1[2]<=28 and ff==0):
728             if (m1[2]==28):
729                 ff = 1
730             p4 = str(m1[2])
731
732         m1 = R3(m1)
733         if (m1[2]<=28 and ff==0):
734             if (m1[2]==28):
735                 ff = 1
736             p5 = str(m1[2])
737
738         m1 = R3(m1)
739         if (m1[2]<=28 and ff==0):
740             if (m1[2]==28):
741                 ff = 1
742             p6 = str(m1[2])
743
744         m1 = R3(m1)
745         if (m1[2]<=28 and ff==0):
746             if (m1[2]==28):
747                 ff = 1
748             p7 = str(m1[2])
749     if(m1[1] in Dias_30):
750         if (m1[2]<=30 and ff==0):
751             if (m1[2]==30):
752                 ff = 1
753             p4 = str(m1[2])
754
755         m1 = R3(m1)
756         if (m1[2]<=30 and ff==0):
757             if (m1[2]==30):
758                 ff = 1
759             p5 = str(m1[2])
760
761         m1 = R3(m1)
762         if (m1[2]<=30 and ff==0):
763             if (m1[2]==30):

```

```

764         ff = 1
765         p6 = str(m1[2])
766
767         m1 = R3(m1)
768         if (m1[2]<=30 and ff==0):
769             if (m1[2]==30):
770                 ff = 1
771                 p7 = str(m1[2])
772     if(m1[1] in Dias_31):
773         if (m1[2]<=31 and ff==0):
774             if (m1[2]==31):
775                 ff = 1
776                 p4 = str(m1[2])
777
778         m1 = R3(m1)
779         if (m1[2]<=31 and ff==0):
780             if (m1[2]==31):
781                 ff = 1
782                 p5 = str(m1[2])
783
784         m1 = R3(m1)
785         if (m1[2]<=31 and ff==0):
786             if (m1[2]==31):
787                 ff = 1
788                 p6 = str(m1[2])
789
790         m1 = R3(m1)
791         if (m1[2]<=31 and ff==0):
792             if (m1[2]==31):
793                 ff = 1
794                 p7 = str(m1[2])
795     elif (c == 4):
796         if(m1[1]== 2):
797             if(bisiesto(m1[0])):
798                 if (m1[2]<=29 and ff==0):
799                     if (m1[2]==29):
800                         ff = 1
801                         p5 = str(m1[2])
802
803                 m1 = R3(m1)
804                 if (m1[2]<=29 and ff==0):
805                     if (m1[2]==29):
806                         ff = 1
807                         p6 = str(m1[2])
808
809                 m1 = R3(m1)
810                 if (m1[2]<=29 and ff==0):
811                     if (m1[2]==29):
812                         ff = 1
813                         p7 = str(m1[2])
814             else:
815                 if (m1[2]<=28 and ff==0):
816                     if (m1[2]==28):
817                         ff = 1
818                         p5 = str(m1[2])
819
820                 m1 = R3(m1)

```

```

821         if (m1[2]<=28 and ff==0):
822             if (m1[2]==28):
823                 ff = 1
824                 p6 = str(m1[2])
825
826                 m1 = R3(m1)
827                 if (m1[2]<=28 and ff==0):
828                     if (m1[2]==28):
829                         ff = 1
830                         p7 = str(m1[2])
831             if(m1[1]in Dias_30):
832                 if (m1[2]<=30 and ff==0):
833                     if (m1[2]==30):
834                         ff = 1
835                     p5 = str(m1[2])
836
837                     m1 = R3(m1)
838                     if (m1[2]<=30 and ff==0):
839                         if (m1[2]==30):
840                             ff = 1
841                             p6 = str(m1[2])
842
843                             m1 = R3(m1)
844                             if (m1[2]<=30 and ff==0):
845                                 if (m1[2]==30):
846                                     ff = 1
847                                     p7 = str(m1[2])
848             if(m1[1]in Dias_31):
849                 if (m1[2]<=31 and ff==0):
850                     if (m1[2]==31):
851                         ff = 1
852                     p5 = str(m1[2])
853
854                     m1 = R3(m1)
855                     if (m1[2]<=31 and ff==0):
856                         if (m1[2]==31):
857                             ff = 1
858                             p6 = str(m1[2])
859
860                             m1 = R3(m1)
861                             if (m1[2]<=31 and ff==0):
862                                 if (m1[2]==31):
863                                     ff = 1
864                                     p7 = str(m1[2])
865         elif (c == 5):
866             if(m1[1]== 2):
867                 if(bisiesto(m1[0])):
868                     if (m1[2]<=29 and ff==0):
869                         if (m1[2]==29):
870                             ff = 1
871                         p6 = str(m1[2])
872
873                         m1 = R3(m1)
874                         if (m1[2]<=29 and ff==0):
875                             if (m1[2]==29):
876                                 ff = 1
877                                 p7 = str(m1[2])

```

```

878         else:
879             if (m1[2]<=28 and ff==0):
880                 if (m1[2]==28):
881                     ff = 1
882                     p6 = str(m1[2])
883
884                     m1 = R3(m1)
885                     if (m1[2]<=28 and ff==0):
886                         if (m1[2]==28):
887                             ff = 1
888                             p7 = str(m1[2])
889             if(m1[1] in Dias_30):
890                 if (m1[2]<=30 and ff==0):
891                     if (m1[2]==30):
892                         ff = 1
893                         p6 = str(m1[2])
894
895                         m1 = R3(m1)
896                         if (m1[2]<=30 and ff==0):
897                             if (m1[2]==30):
898                                 ff = 1
899                                 p7 = str(m1[2])
900             if(m1[1] in Dias_31):
901                 if (m1[2]<=31 and ff==0):
902                     if (m1[2]==31):
903                         ff = 1
904                         p6 = str(m1[2])
905
906                         m1 = R3(m1)
907                         if (m1[2]<=31 and ff==0):
908                             if (m1[2]==31):
909                                 ff = 1
910                                 p7 = str(m1[2])
911         elif (c == 6):
912             if(m1[1]== 2):
913                 if(bisiesto(m1[0])):
914                     if (m1[2]<=29 and ff==0):
915                         if (m1[2]==29):
916                             ff = 1
917                             p7 = str(m1[2])
918                 else:
919                     if (m1[2]<=28 and ff==0):
920                         if (m1[2]==28):
921                             ff = 1
922                             p7 = str(m1[2])
923             if(m1[1] in Dias_30):
924                 if (m1[2]<=30 and ff==0):
925                     if (m1[2]==30):
926                         ff = 1
927                         p7 = str(m1[2])
928             if(m1[1] in Dias_31):
929                 if (m1[2]<=31 and ff==0):
930                     if (m1[2]==31):
931                         ff = 1
932                         p7 = str(m1[2])
933

```

```

934         if (j == 0 or j == 4 or j == 8 or j == 12 or i ==
16 or i == 20):
935             d1 = m1
936             f1 = ff
937             ##             print(d1)
938             ##             print(m1)
939             print('| {:^3}{:~3}{:~3}{:~3}{:~3}{:~3}{:~3}|
'.format(p1,p2,p3,p4,p5,p6,p7), end = "")
940             elif (j == 1 or j == 5 or j == 9 or j == 13 or i
== 17 or i == 21):
941                 d2 = m1
942                 f2 = ff
943                 ##             print(d2)
944                 ##             print(m1)
945                 print('| {:^3}{:~3}{:~3}{:~3}{:~3}{:~3}{:~3}|
'.format(p1,p2,p3,p4,p5,p6,p7), end = "")
946             elif (j == 2 or j == 6 or j == 10 or j == 14 or i
== 18 or i == 22):
947                 d3 = m1
948                 f3 = ff
949                 ##             print(d3)
950                 ##             print(m1)
951                 print('| {:^3}{:~3}{:~3}{:~3}{:~3}{:~3}{:~3}|
'.format(p1,p2,p3,p4,p5,p6,p7), end = "")
952             elif (j == 3 or j == 7 or j == 11 or j == 15 or i
== 19 or i == 23):
953                 d4 = m1
954                 f4 = ff
955                 ##             print(d4)
956                 ##             print(m1)
957                 print('| {:^3}{:~3}{:~3}{:~3}{:~3}{:~3}{:~3}|
'.format(p1,p2,p3,p4,p5,p6,p7))
958                 i += 1
959                 j += 1
960                 x += 1

```