Facultad de Ciencia

Departamento de Matemática y Ciencia de la Computación

Paradigmas de programación

Semestre 2021-01

Ejercicios de ayudantía

Matías Hurtado Carrasco
matias.hurtado@usach.cl

Pablo Pérez-Lantero
8 de mayo de 2021

Indice

1 *<2021-04-29 Thu0* **2**

1.1 Ejemplos 2

2 *<2021-05-06 Thu0* **3**

2.1 Ejercicios 3

2.2 Ejemplos 5

1 *<2021-04-29 Thu>*

1.1 Ejemplos

1. tupla

```
tup = (1 , 2)

for v in tup:
    print(v)
```

2. lista

```
lis = [0, 2, 4, 6]
lis.append(11)

for v in lis:
    print(v//2, v%2)
```

3. funcion

```
def funp(value):
    while value > 0:
        yield value
        value -= 1
    else:
        return 0

for v in funp(5):
    print("funp: " + str(v+1))
```

4. lista de funciones

```
lisf = [lambda x, y: x**2 + y, lambda x, y: x + y**2]

for f in lisf:
    print("f({},{}) = {}".format(2, 3, f(2,3)))
```

5. string

```
st = "hola"

for c in st:
    print("letra {}".format(c))
```

6. conjunto

```
try:
    myset = { "algo", 4 , [1, 2]}
    for e in myset:
        print("{}".format(e))
except TypeError as E:
    print("TypeError:", E)

myset2 = { "algo", 4 , (1, 2)}
for e in myset2:
    print("{} {}".format(e, type(e)))
```

7. diccionarios

```
dic = { "llave1" : "valor1"
        , 2 : 32
        , 4 : "cabello"
        , "9" : 48
       }

dic2 = {"2": 456, 48 : 'casi', 22 : 416, 100 : "no", "que", "queue"}

for k in dic:
    print("dic[{}] = {}".format(k, dic[k]))

for i in range(101):
    if i in dic2:
        print((i, type(i)), (dic2[i], type(dic2[i])))
```

2 *<2021-05-06 Thu>*

2.1 Ejercicios

1. Dado un diccionario, generar listas de tuplas 'llave, valor' ordenada por las llaves y por los valores respectivamente.

```
## para poder comparar strings de tal forma que por ejemplo se cumpla
## "a < B < c < D" (ignorar si es mayuscula) se compara con la key str.lower
## para que las letras mayusculas al momento de comparacion valgan igual que
## las minusculas
dic = {"abc" : 4, "casa" : 16, "ceramica" : 5, "LCC" : 3, "zebra" : 7}

### ordenar por valor de la llave

## usando el comprension de listas que itera sobre las llaves ordenadas
#ld_byKey = [(k, dic[k]) for k in sorted(dic,key=str.lower)]

## pasando a lista las tuplas llave/valor del metodo items() de diccionario,
```

```
## y ordenandola posteriormente
ld_byKey = sorted(list(dic.items()), key=lambda tup : str.lower(tup[0]))

### ordenar por valor de los item 'valor' del diccionario

## usando el comprension de listas que itera sobre las llaves ordenadas con
## respecto a los valores correspondientes
#ld_byVal = [(k, dic[k]) for k in sorted(dic,key=lambda k : dic[k])]

ld_byVal = sorted(list(dic.items()), key=lambda tup : tup[1])

print(ld_byKey)
print(ld_byVal)
```

2. Implementar mergesort para listas con distintos tipos de elementos, y que devuelva un diccionario de los tipos y las listas ordenadas correspondientes que se encuentren.

```
LD = ["Kz3UuwTG", 23, "duGkCWzB", 12,
      "qju7TM5M", "4XnEE9ZJ", 46, "WAk0BSv4",
      32, 14, "76HBLSP8", "IBUs3PZf",
      16, 100, "amMnAkZW", "2jSRNUzD",
      0, 23, "rYBe8CYl", 22]

def mergesort_type(L):
    if (Llen := len(L)) <= 1:
        return dict([(type(x), [x]) for x in L])
    else:
        return merge_type(mergesort_type(L[:(Llen//2)]),
                          mergesort_type(L[(Llen//2):]))

def merge_type(d1, d2):
    D = dict()
    for t in set(list(d1.keys())+list(d2.keys())):
        if t in d1 and t in d2:
            D[t] = merge(d1[t], d2[t])
        elif t in d1:
            D[t] = d1[t]
        elif t in d2:
            D[t] = d2[t]
    return D

def merge(l1, l2):
    L=[]
    while True :
        if (len(l1) == 0):
            L += l2
            break
        if (len(l2) == 0):
```

```
            L += l1
            break
        if (l1[0] < l2[0]):
            L.append(l1[0])
            l1 = l1[1:]
        else:
            L.append(l2[0])
            l2 = l2[1:]
    return L

def merge_add(D,l):
    for obj in l:
        if (tp:=type(obj)) in D:
            D[tp].append(obj)
        else:
            D[tp] = [obj]

for LT in mergesort_type(LD).items():
    print(LT)
```

2.2 Ejemplos

1. Implementacion mediante lambda del factorial. No es muy practica mas alla de ser una implementacion particularmente funcional, pero en python es una solucion bastante forzada y lejos de lo que se esperaria como una solucion.

```
print(
    (lambda x :
     exec('raise TypeError("Only non-negative integers allowed")')
     if type(x) not in (int,float) or x<0 or x%1
     else (
             (fact := (lambda x :
                       1 if x<=1
                       else x * fact(x-1)))(x)))(7)
    )
```