lpii20151

Professor 17:19 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/classificacao-por-quicksort-code-language-python/)

Classificação por quicksort

```
#encoding:utf8
#qpy:console
#
# programa de classificação por quicksort
#
import os
from datetime import datetime
import random
random.seed()
#
#
os.system("clear")

def quicksort(lista):
        L = []
        R = []
        # caso basico
        if len(lista)<=1:
                return lista
        # calcula a chave
        chave = lista[len(lista)/2]
        #
        for i in lista:
                if i < chave:
                        L.append(i)
                if i > chave:
                        R.append(i)
        #finaliza
    return quicksort(L)+[chave]+quicksort(R)
```

```
30
31  # entrada da dados
32  l = input("Digite o tamanho da lista ")
33  A = []
34  i = 0
35  ing = datetime.now()
36  while i<l:
37  #   x = random.randint(1,10*l)
38  #   if x not in A:
39  #       A.append(x)
40  #       i += 1
41      A.append(l-i) # usar essa geração para testar com valores muito altos
42      i += 1
43  fig = datetime.now()
44  ger = fig - ing
45  inicio = datetime.now()
46  B = quicksort(A)
47  fim = datetime.now()
48  duracao = fim - inicio
49  print "-----------"
50  print "Original"
51  print A
52  print "-----------"
53  print "Ordenada "
54  print B
55  print "Tempo (ms) : ","%2.8f"%float(duracao.seconds * 1000 + \
56      float(duracao.microseconds)/1000)
57  print "Tempo (ms) para gerar a lista : ","%2.8f"%float(ger.seconds * 1000 + \
58      float(ger.microseconds)/1000)
```


Professor 17:17 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/classificacao-por-borbulhamento-code-language-python/)

Classificação por borbulhamento

```
1  #encoding:utf8
2  #qpy:console
3  #
4  # programa de classificação por borbulhamento
5  #
6  import os
```

```
from datetime import datetime
import random
#
#
os.system("clear")
random.seed()
# entrada da dados
l = input("Digite o tamanho da lista  ")
A = []
i = 0
while i < l:
    x = random.randint(1,10*l)
    if x not in A:
        A.append(x)
        i += 1
print "------------------------"
print "Lista Original"
print A

perms = True
inicio = datetime.now()
while perms:
    perms = False
        for i in range(len(A)-1):
            if A[i] > A[i + 1]:
               A[i], A[i + 1] = A[i + 1], A[i]
           perms = True

fim = datetime.now()
duracao = fim - inicio
print "-------------"
print "Ordenada"
print A
print "-------------"
print "Tempo (ms) : ","%2.8f"%float(duracao.seconds * 1000 + \
    float(duracao.microseconds)/1000)
```

Professor 17:12 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/ordenamento-por-insercao-code-language-python/)

Classificação por inserção

```
#encoding:utf8
#qpy:console
#
# programa de classificação por inserção
#
import os
from datetime import datetime
import random
random.seed()
#
#
os.system("clear")
# entrada da dados
l = input("Digite o tamanho da lista ")
A = []
# gerando a lista a ser utilizada
i = 0
while i < l:
    x  = random.randint(1,10*l)
    if x not in A:
        A.append(x)
        i += 1

print "-----------------"
print "Lista Original   "
print A
inicio = datetime.now()
for j in range(1, len(A)):
    chave  = A[j]
        i = j - 1
        while A[i] > chave and i >= 0:
            A[i+1] = A[i]
            i -= 1
        A[i+1] = chave
fim = datetime.now()
duracao = fim - inicio
print "----------------"
print "Ordenada "
print A
print "----------------"
print "     (ms) : ","%2.8f"%float(duracao.seconds * 1000 + \
    float(duracao.microseconds)/1000)
```

Professor 17:11 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/ordenamento-por-selecao-code-language-python/)

Ordenamento por seleção.

```
#encoding:utf8
#qpy:console
#
#  Programa de classificação por seleção
#
import os
from datetime import datetime
os.system("clear")
#A = input("Digite a lista a ser ordenada ")
A = []
l = 100000
for i in range(l): A.append(l - i)
print "Lista natural ",A
inicio = datetime.now()
for i in range(len(A)):
    pos_menor = i
    for j in range(i+1,len(A)):
        if A[pos_menor] > A[j]:
            pos_menor = j
    A[i], A[pos_menor] = A[pos_menor],  A[i]
fim = datetime.now()
duracao = fim - inicio
print "Lista Ordenada : ", A
print "Feito em (ms): ","%2.8f" %float(duracao.seconds*1000 + float(duracao.microseconds)/1000)
```

Professor 17:10 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/programa-hanoi-code-language-python-encoding/)

Programa Hanoi.

```
#encoding:utf8
#qpy:console
#
#  programa para calcular os movimentos das torres de hanoi
```

```
5  #
6  import os
7  os.system("clear")
8  #
9  movimentos = 0
10
11 def mover(origem,destino):
12     global movimentos
13     movimentos += 1
14     obj = origem.pop()
15     destino.append(obj)
16     print "-----------"
17     print "1:",h1
18     print "2:",h2
19     print "3:",h3
20
21 def hanoi(n,origem,destino,tmp):
22     if n == 1:
23         mover(origem,destino)
24     else:
25         hanoi(n-1,origem,tmp,destino)
26         mover(origem,destino)
27         hanoi(n-1,tmp,destino,origem)
28 # inicio
29 x = int(raw_input("digite o número de anéis "))
30 h1 = []
31 h2 = []
32 h3 = []
33 for i in range(0,x): h1.append(x-i)
34 print "1:",h1
35 print "2:",h2
36 print "3:",h3
37 hanoi(x,h1,h3,h2)
38 print "Número de movimentos foi", movimentos
39 # fim do programa
```

Professor 17:09 em 23 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/23/implementar-o-mergesort-em-python/)

Implementar o mergesort em python.

Professor 21:24 em 19 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/20/para-o-final-de-semana-implementar-a-solucao/)

Para o final de semana: Implementar a solução do problema das Torres de Hanói.

Professor 16:48 em 19 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/19/programa-36-encontrar-as-permutacoes-code-language-2/)

Programa 36 : Encontrar as permutações.

```
#
# esta funcao retorna uma lista de listas que são permutações da lista original
def permutacoes(lista):
    if len(lista) == 1: # Caso base
        return [lista]
    primeiro = lista[0]
    resto = lista[1:]
    resultado = []
    for perm in permutacoes(resto):
        for i in range(len(perm)+1):
            resultado = resultado + [perm[:i]+[primeiro]+perm[i:]]
    return resultado
# fim da função
x = input("Digite a lista ")
print permutacoes(x)
# fim do programa
```

Gabriel Santos 15:33 em 21 de junho de 2015

Professor eu copiei o código mas ao compilar é acusado o seguinte erro:

TypeError: can only concatenate tuple (not “list”) to tuple

Professor 22:22 em 21 de junho de 2015

O programa não tem erros. Verifique se você entrou com os dados corretamente (Na forma de lista. Ex. [1,2,3]). Falou.

Professor 16:47 em 19 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/19/programa-36-encontrar-as-permutacoes-code-language/)

Programa 36 : Encontrar as permutações.

```
#
# esta funcao retorna uma lista de listas que são permutações da lista original
def permutacoes(lista):
    if len(lista) == 1: # Caso base
        return [lista]
    primeiro = lista[0]
    resto = lista[1:]
    resultado = []
    for perm in permutacoes(resto):
        for i in range(len(perm)+1):
            resultado = resultado + [perm[:i]+[primeiro]+perm[i:]]
    return resultado
# fim da função
x = input("Digite a lista ")
print permutacoes(x)
# fim do programa
```

Professor 17:30 em 16 de junho de 2015

Link Permanente (https://lpii20151.wordpress.com/2015/06/16/exercicio-para-casa-calcular-todas-as-permutacoes-possiveis/)

Exercicio para casa: Calcular todas as permutações possíveis de uma dada lista.