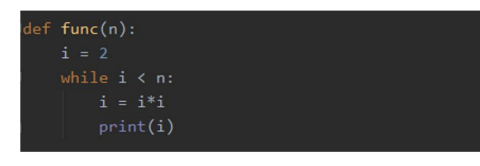
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Is, ==:** | | | | | | | | | | | | **Operations: השוואה** | | | | | | | | | | | | | **String Handling:** | | | | | | | | | |
| isubclass(son,dad) | | | | Assert – מחזיר שגיאה אםFalse | | | | | | | | x + y  X ^ y | | | | | | | \_\_add\_\_(s,o)  x xor y | | | | | | string[1:3] | | | | | | | | tr | |
| isinstance(x,class) raise – מחזיר שגיאה | | | | | | | | | | | | y + x | | | | | | \_\_radd\_\_(s,o) | | | | | | | string[::-1] | | | | | | | | gnirts | |
| Bool (a is b) | | True same place in memory | | | | | | | | | | a + b | | | | | \_\_concat\_\_(s,o) | | | | | | | | string[:-1] | | | | | | | | strin | |
| A==B | | | | True if Same value | | | | | | | | x - y | | | | | | | \_\_sub\_\_(s,o) | | | | | | string[-3:] | | | | | | | | ing | |
|  | | outer is outer **true** | | | | | | | | | | -y | | | | | | | \_\_neg\_\_(s) | | | | | | string[:-3] | | | | | | | | str | |
| outer()is outer() **false** | | | | | | | | | | x \* y | | | | | | | \_\_mul\_\_(s,o) | | | | | | string[-1] | | | | | | | | g | |
| outer()==outer() **false** | | | | | | | | | | x \*\*y | | | | | | | \_\_pow\_\_(s,o) | | | | | | string.find('s') | | | | | | | | 0 | |
| [1,2,(3,4)]==[1,2,(3,4)] | | | | | | | | | | x / y  x==y | | | | | | | \_\_div\_\_(s,o) \_\_eq\_\_(s,y) | | | | | | string.upper()  isalpha, digit, lower | | | | | | | | STRING  האם זה | |
| [1,2,(3,4)]is[1,2,(3,4)] | | | | | | | | | |
| def loadTableAsArray(filename):  \_\_f = open(filename,'r')  \_\_header\_line =f.readline().strip(',\n')  \_\_column\_names = header\_line.split(',')  \_\_Mat,row\_names=[],[]  \_\_for line in f:  \_\_\_\_tokens = line.split(',')  \_\_\_\_row\_names.append(tokens[0])  \_\_\_\_values = [int(n) for n in tokens[1:]]  \_\_\_\_mat.append(values)  \_\_f.close()  \_\_row\_names = np.asarray(row\_names)  \_\_column\_names = np.asarray(column\_names)  \_\_data = np.asarray(mat,dtype=float)  \_\_return data, column\_names,row\_names | | | | | | | | | | | | x % y | | | | | | | \_\_mod\_\_(s,o) | | | | | | string.lower() | | | | | | | | string | |
| x < y | | | | | | | \_\_lt\_\_(s,o) | | | | | | string.replace('str','th') | | | | | | | | | thing |
|  | | | | | | |  | | | | | |  | | | | | | | | |  |
| x <= y | | | | | | | \_\_le\_\_(s,o) | | | | | | string.startswith(x) | | | | | | | | T\F | |
| x > y | | | | | | | \_\_gt\_\_(s,o) | | | | | | '-'.join([a,b,c]) | | | | | | | | 'a-b-c' | |
| x >= y | | | | | | | \_\_ge\_\_(s,o) | | | | | | string.split() | | | | list split by ‘\_’ | | | | | |
| x != y | | | | | | | ­­\_\_ne\_\_(s,o) | | | | | | string.isalpha() | | | | | | is letter? | | | |
| x in y  len(x) | | | | | | | contains(s,o  \_\_len\_\_(s) | | | | | | string.rstrip(x) | | | | | cut x’s from end | | | | |
| **משתנה סטאטי** לפני פונ', לקרוא (classname.var)  **פעולה** **סטאטית** – פעולה רגילה פשוט בתוך המחלקה. | | | | | | | | | | | | | **Random:**from random import random0->1  Uniform(x,y) – float betwn x,y  randrange(from,to,jump)  choice([1,5,6,-44,9])pick one  shuffle([1,2,3])=[3,1,2]  randint(x,y) – int betwn x,y  sample- רשימה ו k איברים ממנה | | | | | | | | | |
| **List Handling tuple:** | | | | | | | | | | **Dictionary Handling:** | | | | | | | | | | | | | **Math (Numpy):** | | | | | | | | | | | |
| list = [a,a,b,a,c,w] | | | | | | | | | | dict = {"k1":v1, "k2":v2} | | | | | | | | | | | | | import numpy as np | | | | | | | | | | | |
| L1 + l2 = | | [l1, l2] | | | | | | | | dict{"k0"}=3 | | | | | | | Sets d[ko] to 3 | | | | | | np.log(8,2) | | | | | | | | Base 8, 2 | | | |
| List[-6:-1:2] | | | | a,b,c | | | | | | K in D | | | | | | | Check if k is key | | | | | | np.sqrt(x) | | | | | | | | | | | |
| list[0:3:2] | | | | [a,b] | | | | | | dict.get(k, def) | | | | | | | If not exist, return def | | | | | | np.loadtxt('fname', delimiter=',') | | | | | | | | | | | |
| lst.insert(3,v) | | | | [a,a,b,v,a,c,w] | | | | | | dict.keys() | | | | | | | List of keys | | | | | | np.arange(1,2.5,0.5) | | | | | | | | | [1,1.5,2] | | |
| lst.append(x) | | | | [a,a,b,v,a,c,w,x] | | | | | | dict.items() | | | | | מחזיר רשימה של טאפל  [(k1,v1),(k2,v2)] | | | | | | | | np.linspace(1,3,5) | | | | | | | | [1,1.5,2,2.5,3] | | | |
| list.extend([y,z]) | | [...,y,z] | | | | | | | | d.copy()  del d[k] | | | | | | | מחזיר העתקה רדודה  מוחק בלי החזרת ערך | | | | | | np.random.rand(3,2) | | | | | | | | מערך של רנד(0-1) בגודל הנתון | | | |
| l.count(a) | | Count a in list | | | | | | | | dict.values() | | | | | | | רשימה של הערכים | | | | | | **sorted**(items to be sorted, cmp = comparator function, key = pre-comparison function, reverse = True\False) יוצר חדש  **sort** ממיין את הקיים | | | | | | | | | | | |
| list.remove(a) | | Remove first 'a' | | | | | | | | dict.pop(k, d) | | | Removes key k & return its val. If not k, return d or KeyError) | | | | | | | | | |
| copy = list[:] | | | | | | | | | | d.update(d2) | | | | | | | Adds d2 items to d | | | | | |
| list.pop(x) | | remove index x | | | | | | | | **Range Handling:**  range(2,5) [2,3,4]  range(0,5,2) [0,2,4]  range(5,0,-2) [5,3,1] | | | | | | | | | | | | | **Fstring**  "Hello, %s. You are %s." % (name, age) | | | | | | | | | | | |
| list.reverse() reverse the list | | | | | | | | | |
| list.sort()בסדר עולה | | | | | | | | | |
| list.index(x)  sorted(lst) | | First ind of x  A | | | | | | | |
| **Stack**  Push(v)  Pop()  Top() | **both**  Is\_empty  Size() | | | | **Queue**  Enqueue(val)  Dqueue() | | | | | **Node**  value  next | | | | | | | **tree**  val  right  left | |  | | | | | | | | Pre – 15,8,3,12,7,13  In – 3,8,12,15,7,13  Post – 3,12,8,13,7,15 | | | | | | | |
| **Numpy:**  x.argmin() | | | | index of smallest value in x | | | | | | | | | | | | | x.hstack(y) | | | | | Put y right to x [xy] | | | | | | | | | | | | |
| np.array([[1,2,3],[4,5,6]]) np.array(list) | | | | | | | | | | | | | | | | | x.vstack(y) | | | Put y under x | | | | | | | | | | | | | | |
| Round– עיגול לקרוב ביותרfloor- למטה siling- למעלה | | | | | | | | | | | | | | | | | axis=0 | | | along vertical (each column) | | | | | | | | | | | | | | |
| x.min\max() | | | min\max values within array | | | | | | | | | | | | | | axis=1 | | | along horizontal (each row) | | | | | | | | | | | | | | |
| x[r1:r2:skip,c1:c2:skip] | | | | | | | | | | | | | | | | | x.shape[0] | | | Rows (including)np.ndim- מימד המערך | | | | | | | | | | | | | | |
| x.sum(axis=0\1) | | | sum of column or row | | | | | | | | | | | | | | x.shape[1] | | | Columns (including) | | | | | | | | | | | | | | |
| np.sum(x) | | | sum of entire array | | | | | | | | | | | | | | x.resize((height including, width including)) | | | | | | | | | | | | | | | | | |
| x.sort(axis=0\1) | | | sort array | | | | | | | | | | | | | | x == y [T,F,T,F,F...] | | | | | | | | | | | | | | | | | |
| np.sort(x,axis=0\1) | | | | | | instance of sorted array | | | | | | | | | | | x[r,:] | | | row r x[:,c] column c | | | | | | | | | | | | | | |
| np.minimum(x,y) | | | array of min values of x and y | | | | | | | | | | | | | | np.newaxis | | | A[:,newaxis] הגדלה במימד | | | | | | | | | | | | | | |
| np.maximum(x,y) | | | array of max values of x and y | | | | | | | | | | | | | | x.any() | | | are any true? | | | | | | | | | | | | | | |
| np.mean(x) | | | mean of entire array | | | | | | | | | | | | | | (x==0).any() | | | does x contain 0? | | | | | | | | | | | | | | |
| x.mean(axis=0) | | | array of mean of each col\row | | | | | | | | | | | | | | x.all() | | | are all true? | | | | | | | | | | | | | | |
| np.median(x) | | | median of entire array | | | | | | | | | | | | | | x.nonzero() | | | get true indx in a dims touple([d1],[d2]) | | | | | | | | | | | | | | |
| x.median(axis=0) | | | array of median of each | | | | | | | | | | | | | | x op y | | | array of op results between x and y | | | | | | | | | | | | | | |
| np.concatenate((x,y),axis=0\1)משרשר עפ כיוון none | | | | | | | | | | | | | | | | | x op a | | | array of op results with a over x | | | | | | | | | | | | | | |
| np.diag(v) | | | diagonal of vector | | | | | | | | | | | | | | x[mask] = c where mask is True, change to c | | | | | | | | | | | | | | | | | |
| x.Transpose | | | Transpose | | | | | | | | | | | | | | x[(x>a)&(x<b)]=c | | | | | | | where condition is met, change to c | | | | | | | | | | |
| x[a,b] | | | value at coordinate | | | | | | | | | | | | | | x[x!=a]= c where value not a, change to c | | | | | | | | | | | | | | | | | |
| x = np.copy(y) | | | make x a copy of y | | | | | | | | | | | | | | x = np.zeros\_like(y) make x a zero-copy of y | | | | | | | | | | | | | | | | | |
| np.vdot(x,y) | | | dot product of x and y @ | | | | | | | | | | | | | | x = np.full\_like(y,val) make x y-shaped and fill with val | | | | | | | | | | | | | | | | | |
| np.bincount() | | | היסטוגרמה של המערך במילון | | | | | | | | | | | | | | np.where(con,T) | | | | | באיזה אינדקסים נמצא האובייקט במערך | | | | | | | | | | | | |
| np.argsort(x) | | | מחזיר רשימה ממוינת עפ אינדקסים | | | | | | | | | | | | | | x.reshape(y,z) | | | | | Change the shape to y\*z | | | | | | | | | | | | |
| x.arange(3)  x.cumsum  x.choose  np.unique(array, return\_index=True\ False) | | | Create [0,1,2]  סוכם לפי סדר מבוקש  בוחר איברים עפ אינדקס ספציפי | | | | | | | | | | | | | | x.flatten()  x.clip  x.partition | | | | | הופך את המטריצה למערך ארוך  חותך מערך עפ מינ, מקס שהכנסתי  חותך מערך עפ אינדקס שהכנסתי | | | | | | | | | | | | |
| **Graphs:**  import matplotlib.pyplot as plt  x = np.linspace(-np.pi,np.pi)  y,z = np.cos(x),np.sin(x)  plt.plot(x,y)  plt.plot(x,z)  plt.show() | | | | | | | | | **Oop:**  Private in constractur:\_\_obj1=OB  heritence more than 1: z(x,y):  For func in 2 fathers - go x,update by y  If z hasn’t the func, checks if exist in x, if not checks in y  תכונות פרטיות לא עוברות אוטומטית  Return mutable objects with copy, that the user cant change | | | | | | | | | | | |  | | | | | | | | | | | | | |
| **Image Handling:**  import matplotlib.pyplot as plt  from scipy import misc  image = misc.imread('pic.jpg')  misc.imsave('new.bmp',image)  image.rotate(deg) | | | | | | | | |
| **Show Image:**  plt.figure() 255-לבן 0-שחור  plt.imshow(image) erosion לבן הופך לשחור  plt.show() dilation שחור הופך ללבן | | | | | | | | | | | | | | | | **lambda**  a=[[3,18],[2,25],[1,9],[7,11]]  print(sorted(a,key=lambda x:x[1]))  [[1, 9], [7, 11], [3, 18], [2, 25]] | | | | | | | | | |
| **def Heatmap**(data, rowLab, colLab, title, xlab, ylab):  plt.imshow(data,interpolation='None')  plt.title(title)  plt.colorbar()  plt.xlabel(xlab)  plt.xticks(range(len(columnLab)), columnLab)  plt.ylabel(ylabel)  plt.yticks(range(len(rowLabels)),rowLabels)  plt.show() | | | | | | | | | | | | | | | | | |  |  |  | | --- | --- | --- | | סוג המבנה | insert | Find | | רשימה לא ממוינת | Append o(1) | O(n) | | רשימה ממוינת | O(n) | O(logn) | | רשימה מקושרת | הכנסה להתחלה o(1) | O(n) | | רשימה מקושרת ממוינת | O(n) | O(n) | | עץ חיפוש בינארי | גרוע o(n) עץ מאוזן o(logn) | |   **סיבוכיות מקום:**  Copy לא משנה, l1 +=l2, append, extend. משנה: l1+l2, slicing | | | | | | | | | | | | | | | | | |
| **Denoising:**  defmorphological(im, operator= (*min,max)*, nx= 5, ny=5 ):  width, height = im.shape  out\_im= np.zeros(im.shape)  for y in range(height):  for x in range(width):  nlst= neighbours(im, x, y, nx, ny)  out\_im[x, y] = operator(nlst)  return out\_im | | | | | | | | | | | | | | | | | | | **Squeeze Image:**  def squeeze\_image(im,factor):  \_\_\_new\_n = im.shape[0]  \_\_\_new\_m = im.shape[1] / factor  \_\_\_new\_mat = np.zeros((new\_n,new\_m))  \_\_\_for j in range(new\_mat.shape[1]):  \_\_\_\_\_\_curr=range(j\*factor,min((j+1)\*factor,im.shape[1]))  \_\_\_\_\_\_new\_mat[:,j] = im[:,curr].mean(axis=1)  \_\_\_return new\_mat | | | | | | | | | | | | | | | |
| גוונים | | | | | | | | | | | | | | | | | | | x = lambda x: lambda y:(x,y) | | | | | | | | | **זמני ריצה:**  ממוצע – רק מילון, O(1)  עץ, פיבונאצי, האנוי – 2^n  **Box-Plot:**  Plt.boxplot(data)->plt.show | | | | | | |
| **Error Handling:**  raise errortype('text')  Name error  Syntac error  IoError  ValueError  TypeError  ZeroDivisionError  IndexError  OverflowError  try:  \_\_\_  except exception as err1:  \_\_\_print err1.args[0]  finally: #runs anyway  \_\_\_  Assert(cond)  if false make error | | | | | | | **File Handling:**  file = open('file.txt', 'w\r\a')  file.read()  for line in file  file.readline()  file.readlines()  file.close()  file.write(str)  '\n' - new line  **Sort:**  Insertion sort-(n^2)החלפת משתנה והזזת הגדול  Heap sort- עפ עץ בינארי o(n^2) | | | | | | | | | **Cumsum with list comp**  lst=[4,5,6,7,8,]  cumsum=[sum(lst[:i+1]) for i in range (len(lst))]  [4, 9, 15, 22, 30]  cumsum=[sum(lst[:i+1]) for i in range (len(lst)) if lst[i]>6]  [22,30]  **ליסט קומפריהנסון עם 2 לולאות = 2 סוגריים מרובעים** | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | **O(n)** | | | | | | | | |
| **Search & Sort:**  Binary Search - in sorted list - if value<middle, search lower half of list.**o(logn)**  Bubble Sort - compare sequentially and swap if needed  (**best o(n),worst/average o(n^2)**  Merge Sort - split in two until length of one, sort each by parts, merge and sort.**o(nlogn)**  \*quick sort - בוחרים איבר ציר רנדומלי X.  ממינים את הרשימה לשתי רשימות (גדול מX קטן מX)ממשיכים ברקורסיה o(nlogn) or worst case o(n^2) | | | | | | | | | | | | | | | |
| **Bubble Sort:o(n^2)**  def bubble\_sort(lst):  \_\_\_for i in range(len(lst)-1, 0, -1):  \_\_\_\_\_\_for j in range(i):  \_\_\_\_\_\_\_\_\_if lst[j] > lst[j+1]:  \_\_\_\_\_\_\_\_\_\_\_\_lst[j], lst[j+1]= lst[j+1], lst[j] | | | | | | | | | | | Big O Notation Cheat Sheet | What Is Time & Space Complexity?**quick Sort:**  def quick\_sort(lst):  if len(lst) <= 1:  return lst  # use last number as pivot  pivot = lst[-1]  left\_lst = []  right\_lst = []  for i in range(len(lst)-1):  if lst[i] < pivot:  left\_lst.append(lst[i])  --------else: \_\_\_\_\_\_\_\_\_\_right\_lst.append(lst[i])  return quick\_sort(left\_lst) + [pivot] + quick\_sort(right\_lst) | | | | | | | | | def is\_BST(root):  stack = []  prev = None  while root or stack:  while root:  stack.append(root)  root = root.left   root = stack.pop()  if prev and root.val < prev.val   return False  prev = root  root = root.right return True | | | | | | | | | | | | | | |
| **משתנה איטרטיבי – ממומשות לו:**  **\_\_iter\_\_(self)**  **\_\_next\_\_(self)**  **בשביל לרוץ על משתנה,צריך פונ’ השוואה** | | | | | | | | | | |
| **פריצה למשתנה פרטי:**  **לטובת "מרגל"- random** | | | | | | | | | | |
| **Histogram:**  def histogram(string):  \_\_\_d = { }  \_\_\_for char in string:  \_\_\_\_\_\_count = d.get(char,0)  \_\_\_\_\_\_d[char] = count + 1  תמונה שמכילה טקסט, צילום מסך, גופן, מספר  התיאור נוצר באופן אוטומטי\_\_\_return d | | | | | | | | **Fibonacci w\ Memo:**  def fib(n,mem=None):  \_\_\_if n<2:  \_\_\_\_\_\_return n  \_\_\_if mem == None:  \_\_\_\_\_\_mem = {}  \_\_\_if n not in mem: | | | | | | \_\_\_\_\_\_mem[n] =  fib(n-1,mem)+fib(n-2,mem)  \_\_\_return mem[n] | | | | | |
| **General knowledge:**  **Math series: n(a1+an)/2** | | | | | | | | **Time complexity:** | | | | | |  | | | | | | | | | | | | | | | | | | | | |

A computer screen with white text

Description automatically generatedתמונה שמכילה טקסט, צילום מסך, מסמך, גופן

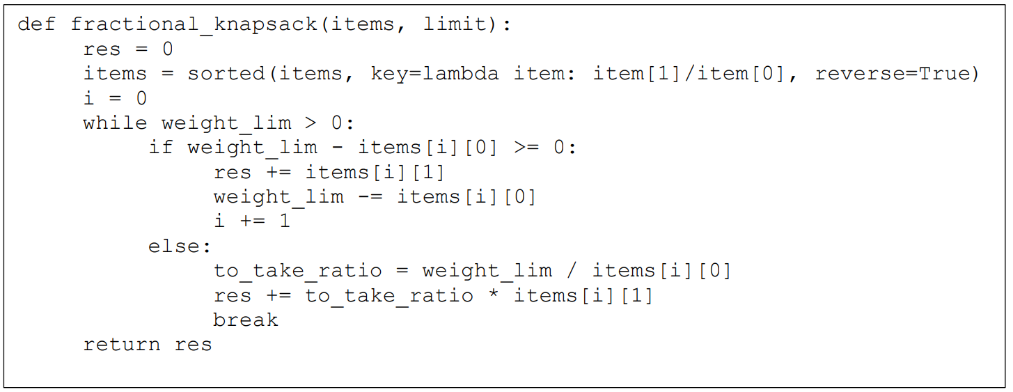
התיאור נוצר באופן אוטומטי

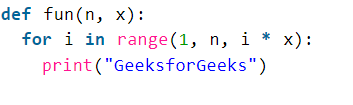
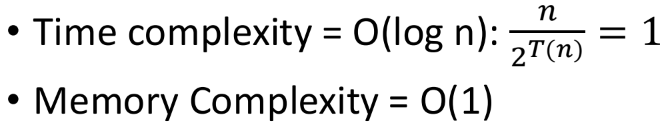


תמונה שמכילה טקסט, גופן, צילום מסך

התיאור נוצר באופן אוטומטיA computer screen with text and images

Description automatically generated





A screenshot of a computer screen

Description automatically generatedתמונה שמכילה טקסט, צילום מסך, גופן, מספר

התיאור נוצר באופן אוטומטי

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer program

Description automatically generated