

Taks-1

```
# input:1 10
# hint :9+1 9+3 9+5 9+7 9+9 9+11 ...
# ouput:10 12 14 16 18 20 22 24 26 28
```

Task-2

```
# input:5
# hint 5^1+1^1 5^2+2^2 5^3+3^3
# output:6 29 152 881 6250
```

Task-3

```
# st="ECE4EEE3CIV2CSE4"
# ouput:ECEECEEECE
#      EEEEEEEEE
#      CIVCIV
#      CSECSECSECSE
```

In [6]:

```
1 #Task-3
2 # st="ECE4EEE3CIV2CSE4"
3 # ouput:ECEECEEECE
4 #      EEEEEEEEE
5 #      CIVCIV
6 #      CSECSECSECSE
7
8 st="ECE4EEE3CIV2CSE4"
9 branches=[]
10 values=[]
11 es=''
12 for evry_chr in st:
13     if evry_chr.isalpha():
14         es=es+evry_chr
15     else:
16         branches.append(es)
17         values.append(evry_chr)
18         es=''
19 print(branches)
20 print(values)
21
22
```

```
['ECE', 'EEE', 'CIV', 'CSE']
['4', '3', '2', '4']
```

```
In [7]: 1 for i in range(len(branches)):
        2     print(branches[i]*int(values[i]))
```

```
ECEEECEEECE
EEEEEEEEE
CIVCIV
CSECSECSECSE
```

```
In [8]: 1 for i in range(len(branches)):
        2     r=branches[i][::-1]
        3     print(r*int(values[i]))
```

```
ECEEECEEECE
EEEEEEEEE
VICVIC
ESCESCESCESC
```

```
In [19]: 1     # input:1 10
        2     # hint :9+1 9+3 9+5 9+7 9+9 9+11 ...
        3     # ouput:10 12 14 16 18 20 22 24 26 28
        4     a=9
        5     sum=0
        6     for i in range(1,20):
        7         if i%2!=0:
        8             sum=a+i
        9             print(sum,end=" ")

10 12 14 16 18 20 22 24 26 28
```

```
In [17]: 1 # Task-2
        2     # input:5
        3     # hint 5^1+1^1 5^2+2^2 5^3+3^3
        4     # output:6 29 152 881 6250
        5     i=int(input())
        6     for j in range(1,10):
        7         print(i**j+j**j)
```

```
5
6
29
152
881
6250
62281
901668
17167841
389373614
```

Dictionary

- Dictionary is Collection of Hetrogenous Data Set
 - ex:chr,int,spl char,..etc
 - It is an mutale (we can read elements,remove,modify)

- Here Data in form of Key and value pair
- All keys are unique but not values
- representation is data type is dict
- we can insilize the dict with {}

```
In [20]: 1 d={"key 1":'value1',"Key 2":"Value2","key 3":"Value3"}
```

```
In [22]: 1 d1={"cbit":2008,"python":1989,"apssdc":2014}
        2 d1
```

```
Out[22]: {'cbit': 2008, 'python': 1989, 'apssdc': 2014}
```

```
In [23]: 1 print(dir(dict))

['_class__', '__contains__', '__delattr__', '__delitem__', '__dir__', '__doc__',
 '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__gt__',
 '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__',
 '__lt__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__',
 '__setattr__', '__setitem__', '__sizeof__', '__str__', '__subclasshook__', 'clear',
 'copy', 'fromkeys', 'get', 'items', 'keys', 'pop', 'popitem', 'setdefault',
 'update', 'values']
```

```
In [24]: 1 #Geeting all keys
        2 d1.keys()
        3
```

```
Out[24]: dict_keys(['cbit', 'python', 'apssdc'])
```

```
In [25]: 1 #values
        2 d1.values()
```

```
Out[25]: dict_values([2008, 1989, 2014])
```

```
In [26]: 1 d1['Vbit']=2008 #Adding New elements in to dict
```

```
In [27]: 1 d1
```

```
Out[27]: {'cbit': 2008, 'python': 1989, 'apssdc': 2014, 'Vbit': 2008}
```

```
In [28]: 1 d1['python']=1991 #Updating dict value
```

```
In [29]: 1 d1
```

```
Out[29]: {'cbit': 2008, 'python': 1991, 'apssdc': 2014, 'Vbit': 2008}
```

```
In [30]: 1 d1['vbit']=2000
```

```
In [31]: 1 d1
```

```
Out[31]: {'cbit': 2008, 'python': 1991, 'apssdc': 2014, 'Vbit': 2008, 'vbit': 2000}
```

```
In [33]: 1 d1.pop('vbit')#Removing paticular Element from Dict
```

```
Out[33]: 2000
```

```
In [34]: 1 d1
```

```
Out[34]: {'cbit': 2008, 'python': 1991, 'apssdc': 2014, 'vbit': 2008}
```

```
In [35]: 1 d1.popitem()
```

```
Out[35]: ('vbit', 2008)
```

```
In [37]: 1 d1
```

```
Out[37]: {'cbit': 2008, 'python': 1991, 'apssdc': 2014}
```

```
In [38]: 1 #Displaying all the Keys and Values wihch present in the dict
2
3
4 # cbit-->2008
5 # python-->1991
6 # apssdc-->2014
7 for k,v in d1.items():
8     print(k,"-->",v)
```

```
cbit --> 2008
python --> 1991
apssdc --> 2014
```

```
In [ ]: 1 #Tasks : Using Functions
2         #1.Create a contact dict.Then add the contact details
3         #like name and number to dict
4         #note:if name in dict disply name already exist
5         #2.Update the Contact which present in the dict
6         #note:if not name in dict disply name not exist
7         #3.Delete the contact name and number from dict
8         #note:if not name in dict disply name not exist
9         #4.List out the all contacts from dict
10        #Note:if contacts is empty disply "empty List"
11
```

```
In [39]: 1 contacts={}
```

```
In [ ]: 1 #1.Create a contact dict.Then add the contact details
2         #like name and number to dict
3         #note:if name in dict disply name already exist
```

```
In [59]: 1 #Funtion to add a contact to dict
2 def addContact(name,number):
3     if name not in contacts:
4         contacts[name]=number
5         print("Contact added Successfully.")
6     else:
7         print("Already name Exist:",name)
8
9 name=input("enter the Name")
10 number=int(input("Enter the number"))
11 addContact(name,number)
```

enter the Namehanuman
Enter the number1234
Contact added Successfully.

```
In [47]: 1 contacts
```

```
Out[47]: {'muni': 111111, 'vijay': 4544}
```

```
In [50]: 1 #Funtion to update a contact to dict
2 def updatecontact(name):
3     if name in contacts:
4         number=int(input("Enter the number"))
5         contacts[name]=number
6         print("Contact updated Successfully.")
7     else:
8         print("Not avalilabe :",name)
9
10 name=input("enter the Name")
11
12 updatecontact(name)
```

enter the Namemuni
Enter the number3333333
Contact updated Successfully.

```
In [55]: 1 #Funtion to update a contact to dict
2 def deleteContact(name):
3     if name in contacts:
4         contacts.pop(name)
5         print("deleted Successfully.")
6     else:
7         print("Not avalilabe :",name)
8
9 name=input("enter the Name")
10
11 deleteContact(name)
```

enter the Namevijay
deleted Successfully.

In [56]: 1 contacts

Out[56]: {}

```
In [60]: 1 #Funtion to update a contact to dict
2 def displayAllContacts(d):
3     if len(contacts)<=0:
4         print("Empty contacts list ")
5     else:
6         for k,v in contacts.items():
7             print(k,"-->",v)
8     displayAllContacts(contacts)
```

hanuman --> 1234

```
In [ ]: 1 #Task:
2     # 1.
3     #     input: {'cse':5,"ece":4,"eee":3,"civ":2,"mech":1}
4     #     output:csecsecsecsecse
5     #             eceeeceeece
6     #             eeeeeeeeee
7     #             civciv
8     #             mech
9     # 2.
10    # Find the Frequency char and count
11    #input:data='weareincbitproddaturLearingpythonprogramming'
12    #     data1="hihellocbitccccccc"
13    #     o/p:c-->8
```

```
In [64]: 1 #Task:
2     # 1.
3     #     input: {'cse':5,"ece":4,"eee":3,"civ":2,"mech":1}
4     #     output:csecsecsecsecse
5     #             eceeeceeece
6     #             eeeeeeeeee
7     #             civciv
8     #             mech
9     b={'cse':5,"ece":4,"eee":3,"civ":2,"mech":1}
10    v=list(b.values())
11    v
```

Out[64]: [5, 4, 3, 2, 1]

```
In [69]: 1 for k,v in b.items():
2         print(k*v)
```

csecsecsecsecse
eceeeceeece
eeeeeeeee
civciv
mech

[illegible]
$$r \rightarrow 6$$

```
In [3]: 1 # 8 6 1
        2 c=list(map(int,input().split()))
        3 sorted(c)
```

8 6 1

Out[3]: [1, 6, 8]

```
In [ ]: 1 # 5
        2 # 85 25 65 21 84
        3      55514%10
        4      no
```

```
In [7]: 1 n=int(input())
        2 c=list(map(int,input().split()))
        3
```

75841

679 29980 47815 72976 87375 53880 15953 45632 18034 48300 93609 21568 51752 8
4044 37900 95328 75110 90176 49575 99600 96008 76128 58868 12864 26875 80788
65666 32928 80792 65400 78014 3232 75006 11348 2950 73728 8928 9576 42511 888
00 66879 8772 4940 91456 83800 53404 70883 16928 84865 45000 83283 79008 6968
0 96040 2425 44224 12846 78284 43267 91200 42298 22736 37444 67616 98400 4062
8 11521 39344 27747 58800 84043 72480 65880 97252 23750 87088 69982 63628 469
53 51200 1035 680 46088 68384 85600 88024 12111 66048 67136 24000 35743 8064
96845 59612 4125 10272 84212 52792 71657 40000 51026 44308 53837 8320 57750 4
2864 54975 58704 27129 33500 65096 70432 84587 83928 35950 21440 53536 880 75
998 12800 89755 63116 23038 56960 81250 80664 70547 63776 8578 9600 82342 237
44 23228 97100 25900 47488 78512 33064 32006 0 39921 45868 98694 67968 94250
61756 43428 50384 43583 72500 31888 65952 97462 36632 10600 97056 91855 77992
2284 64800 54739 43112 93856 75600 16425 19708 40082 4864 7135 6800 83605 755
68 86606 7196 18125 86944 48910 88632 71195 50400 9765 12092 84075 36544 4650
54656 15621 34160 72942 47300 55505 25664 70510 18072 59775 36416 21178 41096
69477 0 95017 17728 67074 54688 24225 22636 14371 30464 94529 9700 6811 50176
63354 39148 25 33888 76398 5732 6820 22800 99367 47856 87882 18048 86875 4951
6 62818 77440 57182 65800 96350 80832 25624 86944 67150 39088 40617 13116 668
86 48888 37825 71812 18188 66656 8775 26152 77276 26118 50562 62588 75617 515

In [5]: 1 99999**9999

Out[5]: 90484601407448825888563042346605458233021361015111900453967466926159666809167
00718088170993764595735005466808681728335092710027876668181814574169825632603
01181328945280913808154404122478097933042855341097524989622580220720601779294
08355517051542779870390872753410228610521612708564422777676240381647123612979
21095602588297182044601406293891126097503384262397392758903381885708407315590
03274297169335288683147152468015033748902873860329415745229861582631003510081
65511968631748030017931369754756297517055404898589682552909580559196925077781
25080640538116001263237351980942648046469067347141452062971066041835574228224
29902329128094156870198330025829956707465082815062727611590866983450962149416
66579714218254203784311955275862329798231566732490865504650468095023956038929
61827077724209815938107863710387117606954454096483737172916377402752197142812
06187156813736352377124841711073982981205859978489187615781330994626890734977
41596658783625517605328673434351161700233707608319307468775312285241248164136
59917826424776784101527487473669995667555305406897737239655238316886108824355
92473225005699427111953117238328582073798055121446993318632781093018068027807
69836671012896053677213239607356472466990544653769121047171899702592693092425
16375491164648872176321119171549029519906389016720004659235604516291639850619
41776327973467564880884717813768259694996572421552078352193781757787615950895
37884803173117690848711068024198783477915164996711515065512506247466078757068
5228521222116600226202780162556674020477341140502716006522016100714081101022

In [1]: 1 10**9

Out[1]: 1000000000

In [20]: 1 n=int(input())
2 c=list(map(int,input().split()))
3 c

5
85 25 65 21 84

Out[20]: [85, 25, 65, 21, 84]

In [27]: 1 n=int(input())
2 n1=list(map(int,input().split()))
3 re=[]
4 for i in range(len(n1)):
5 remi=n1[i]%10
6 re.append(remi)
7 print(re)
8 for j in range(len(re)):
9 r=re[j]
10 if r%10==0:
11 print("Yes")
12 else:
13 print("No")

5
85 25 65 21 84
[5, 5, 5, 1, 4]
No

Regular Expressions

- It is one language .It will match the some sequence of patterns
- It will returns the 100% output is accurate
- Short form re
- we need to import the re(regular expressions)

In [29]:

```
1 import re
2
3 print(dir(re))
```

```
['A', 'ASCII', 'DEBUG', 'DOTALL', 'I', 'IGNORECASE', 'L', 'LOCALE', 'M', 'MULTI
LINE', 'Match', 'Pattern', 'RegexFlag', 'S', 'Scanner', 'T', 'TEMPLATE', 'U',
'UNICODE', 'VERBOSE', 'X', '_MAXCACHE', '__all__', '__builtins__', '__cached_
__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__spec__',
'__version__', '_cache', '_compile', '_compile_repl', '_expand', '_locale', '_p
ickle', '_special_chars_map', '_subx', 'compile', 'copyreg', 'enum', 'error',
'escape', 'findall', 'finditer', 'fullmatch', 'functools', 'match', 'purge', 's
earch', 'split', 'sre_compile', 'sre_parse', 'sub', 'subn', 'template']
```

match:Will return the only first char match

Search:If get one success it will terminate

FindAll:It will return all cases

In [32]:

```
1 re.match('a', 'apssdc')
```

Out[32]: <re.Match object; span=(0, 1), match='a'>

In [33]:

```
1 re.match('a', 'pssdca')
```

In [34]:

```
1 re.search('s', 'apssdc')
```

Out[34]: <re.Match object; span=(2, 3), match='s'>

In [35]:

```
1 re.search('s', 'sapssdc')
```

Out[35]: <re.Match object; span=(0, 1), match='s'>

In [36]:

```
1 #Findall
2 re.findall('s', 'apssdc')
```

Out[36]: ['s', 's']

symbols and meanings

1. ^-->Exactly starting letter with
2. \$-->Exactly ending with
3. []-->data set
4. {}-->range Set

5. .--->one char/digt/splChar ...etc
6. |--->either this or that
7. • --> with out null
8. *--->multiply

In []:

```

1  #s=cbit1234&^JHGJHG
2  # Get the all digits
3  #Mobile number validation
4      #case:1-->Starting with 6 or 7 or 8 or 9
5      #^[6-9][0-9]{9}$
6      #case:2-->Starting with 0(zero)
7      #^[0][6-9][0-9]{9}$
8      #Case 3:starting with +91
9      #^[+][9][1][6-9][0-9]{9}$
10 #Final Pattern for Mobile number validation
11 #^[+][9][1][6-9][0-9]{9}$|^[0][6-9][0-9]{9}$|^[6-9][0-9]{9}$

```

In [46]:

```

1  li=['9876543210','652434247','1213713501','1327531122','0987654321']
2  pattern='^[+][9][1][6-9][0-9]{9}$|^[0][6-9][0-9]{9}$|^[6-9][0-9]{9}$'
3  for i in li:
4      if re.match(pattern,str(i)):
5          print(i)

```

9876543210

In [45]:

```

1  import re
2  n=input()
3  pattern='^[+][9][1][6-9][0-9]{9}$|^[0][6-9][0-9]{9}$|^[6-9][0-9]{9}$'
4  if re.match(pattern,n):
5      print("yes valid number",n)
6  else:
7      print("Invalid number goto home")

```

131

Invalid number goto home

In []:

```

1  #Task for Email id validation
2      #muni.apssdc@gmail.com
3      #muneiah.t@apssdc.in
4      #example123@gmail.com
5      #student@ksrm.org
6      #student@rguktrkv.ac.in
7      #example@yahoo.com
8      #example@hotmail.com
9      #muneiah@outlook.com
10

```