### 07 - Functions

			Abundant Number
Register N	lo.:		Name:
Ex. No.	:	7.1	Date:

An abundant number is a number for which the sum of its proper divisors is greater than the number itself. Proper divisors of the number are those that are strictly lesser than the number.

### **Input Format**:

Take input an integer from stdin

#### **Output Format:**

Return Yes if given number is Abundant. Otherwise, print No

#### Example input:

12

#### Output:

Yes

Explanation

The proper divisors of 12 are: 1, 2, 3, 4, 6, whose sum is 1 + 2 + 3 + 4 + 6 = 16. Since sum of proper divisors is greater than the given number, 12 is an abundant number.

#### **Example input:**

13

#### Output:

No

#### **Explanation**

The proper divisors of 13 is: 1, whose sum is 1. Since sum of proper divisors is not greater than the given number, 13 is not an abundant number.

For example:

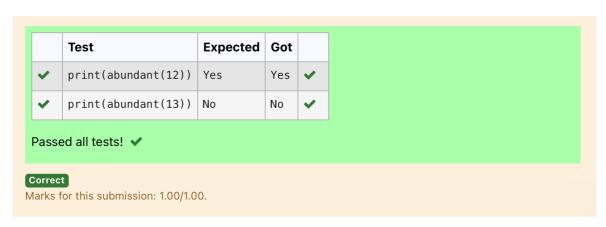
Test Result print(abundant(12)) Yes print(abundant(13)) No

### Program:

def abundant(n):

```
1,s=[],0
for i in range(1,int(n/2)+1):
  if(n\%i==0):
```

```
l.append(i)
for i in l:
    s+=i
if(s>n):
    return("Yes")
else:
    return("No")
```



Ex. No. : 7.2 Date:

Register No.: Name:

.

### Automorphic number or not

An automorphic number is a number whose square ends with the number itself. For example, 5 is an automorphic number because 5\*5=25. The last digit is 5 which same as the given number.

If the number is not valid, it should display "Invalid input".

If it is an automorphic number display "Automorphic" else display "Not Automorphic".

Input Format:

Take a Integer from Stdin

**Output Format:** 

Print Automorphic if given number is Automorphic number, otherwise Not Automorphic

Example input: 5 Output: Automorphic Example input: 25 Output: Automorphic

Example input: 7 Output: Not Automorphic

For example:

Test Result

### Program:

a = str(n\*n)

```
def automorphic(n):
```

```
if(int(a[-1])==n):
```

return("Automorphic")

else:

return("Not Automorphic")

	Test	Expected	Got	
~	<pre>print(automorphic(5))</pre>	Automorphic	Automorphic	~
~	<pre>print(automorphic(7))</pre>	Not Automorphic	Not Automorphic	~

Passed all tests! 🗸

### Correct

Marks for this submission: 1.00/1.00.

Ex. No. : 7.3		Date:
Register No.:		Name:
-	Check	Product of Digits
Write a code to check wo of digits at odd place of Input Format:	_	roduct of digits at even places is divisible by sum e integer.
Take an input integer f	from stdir	1.
Output Format:		
Print TRUE or FALSE		
Example Input:		
1256		
Output:		
TRUE		
Example Input:		
1595		
Output:		
FALSE		
For example:		
Test	Result	
print(productDigits(1256))	True	

# Program:

 $def \ product Digits(n):$ 

print(productDigits(1595)) False

```
a=str(n)
s,p=0,1
for i in range(0,len(a),2):
    s+=int(a[i])
for i in range(1,len(a),2):
    p*=int(a[i])
if(p%s==0):
    return("True")
else:
    return("False")
```



Ex. No. : 7.4 Date:

Register No.: Name:

.

### **Christmas Discount**

An e-commerce company plans to give their customers a special discount for Christmas.

They are planning to offer a flat discount. The discount value is calculated as the sum of all the prime digits in the total bill amount.

Write an python code to find the discount value for the given total bill amount.

#### **Constraints**

```
1 \le \text{orderValue} \le 10e^{100000}
```

Input

The input consists of an integer orderValue, representing the total bill amount.

Output

Print an integer representing the discount value for the given total bill amount.

Example Input

578

Output

12

#### For example:

Test	Result
print(christmasDiscount(578))	12

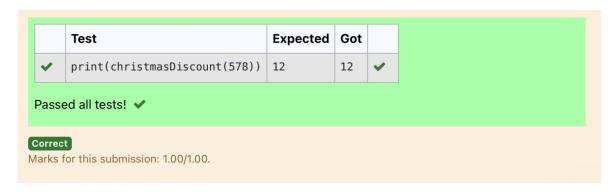
## Program:

def christmasDiscount(n):

```
res=0
while n!=0:
rem=n%10
flag=0
for i in range(1,rem+1):
if rem%i==0:
```

```
flag+=1
if flag==2:
res=res+rem
n=n//10
```

return res



Ex. No.	:	7.5	Date:
Register N	No.:		Name:
			Coin Change
-		-	ent coin change making problem i.e. finding the minimum
			enominations that add up to given amount of money.
		coins are	of values 1, 2, 3, 4
Input Forn	nat:		
Integer inp	out from	ı stdin.	
Output For	rmat:		
return the	minimu	ım numbe	er of coins required to meet the given target.
Example I	nput:		
16			
Output:			
4			
Explanation	n:		
We need or	nly 4 coi	ins of valu	e 4 each
Example I	nput:		
25			
Output:			
7			
Explanation	n:		
We need 6	coins of	f 4 value, a	and 1 coin of 1 value
Progra	am:		
def coinC	hange(	(amount)	:
# Availa	able co	in denon	ninations
coins =	[1, 2, 3]	, 4]	
# Initia from 0 to			re the minimum number of coins for each amount int

dp = [float('inf')] \* (amount + 1)

```
dp[0] = 0 # Base case: 0 coins needed to make amount 0

# Iterate through all amounts from 1 to the target amount
for i in range(1, amount + 1):
    # Iterate through all available coin denominations
    for coin in coins:
        # If the current coin denomination is less than or equal to the
current amount
        if coin <= i:
            # Update dp[i] to be the minimum between its current value
and dp[i - coin] + 1
            dp[i] = min(dp[i], dp[i - coin] + 1)

# The result is stored at dp[amount]
return dp[amount]
amount = int(input())
print(coinChange(amount))</pre>
```



Ex. No. : 7.6 Date:

Register No.: Name:

**Difference Sum** 

Given a number with maximum of 100 digits as input, find the difference between the sum of odd and even position digits.

Input Format:

Take a number in the form of String from stdin.

**Output Format:** 

Print the difference between sum of even and odd digits

Example input:

1453

Output:

1

Explanation:

Here, sum of even digits is 4 + 3 = 7 sum of odd digits is 1 + 5 = 6.

8 ...

Note that we are always taking absolute difference

## Program:

s=sum(b)

Difference is 1.

```
def differenceSum(n):
```

```
a=[]
b=[]
k=str(n)
for i in range(len(k)):
    if int(i)%2==0:
        a.append(int(k[i]))
    else:
        b.append(int(k[i]))
```

```
r=sum(a)
j=s-r
return j
```



Ex. No. : 7.7 Date:

Register No.: Name:

.

### Ugly number

A number is considered to be ugly if its only prime factors are 2, 3 or 5.

[1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, ...] is the sequence of ugly numbers.

Task:

complete the function which takes a number n as input and checks if it's an ugly number. return ugly if it is ugly, else return not ugly

Hint:

An ugly number U can be expressed as:  $U = 2^a * 3^b * 5^c$ , where a, b and c are nonnegative integers.

### For example:

Test	Result
print(checkUgly(6))	ugly
print(checkUgly(21))	not ugly

## Program:

```
def checkUgly(n):
```

```
for i in range(n):
    for j in range(n):
        if (n == (2**i) + (3**j) + (5**k)):
            return("ugly")
    return("not ugly")
```

	Test	Expected	Got	
~	<pre>print(checkUgly(6))</pre>	ugly	ugly	~
~	<pre>print(checkUgly(21))</pre>	not ugly	not ugly	~

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.