1. Check whether a number is positive, negative, or zero

```
def sign_of_number(n):
    if n>0:
        return 'Positive'
    elif n<0:
        return 'Negative'
    else:
        return 'Zero'
```

2. Check whether a number is even or odd

```
def is_even(n):
    return n%2==0
```

3. Check if a given year is a leap year or not

```
def is_leap_year(year):
    return (year%4==0 and year%100!=0) or (year%400==0)
```

4. Find the greatest of two numbers

```
def max_of_two(a,b):
    return a if a>=b else b
```

5. Check whether a person is eligible to vote (age >= 18)

```
def can_vote(age):
    return age>=18
```

6. Check whether a given character is a vowel or consonant

```
def is_vowel(ch):
    return ch.lower() in 'aeiou'
```

7. Check if a number is divisible by 5

```
def div_by_5(n):
    return n%5==0
```

8. Determine whether a given number is single-digit, two-digit, or more than two-digit

```
def digit_category(n):
    n=abs(n)
    if n<10:
        return 'single-digit'
    elif n<100:
        return 'two-digit'
    else:
        return 'more than two-digit'
```

9. Check whether a student has passed or failed (passing marks = 40)

```
def pass_fail(marks):
    return 'Pass' if marks>=40 else 'Fail'
```

10. Find whether the entered number is a multiple of both 3 and 7

```
def mult_of_3_and_7(n):
    return n%3==0 and n%7==0
```

1. Find the greatest among three numbers

```
def max_of_three(a,b,c):
    return max(a,b,c)
```

2. Classify a person based on age: Child (<13), Teenager (13-19), Adult (20-59), Senior (6

```
def classify_age(age):
    if age<13:
        return 'Child'
    elif age<=19:
        return 'Teenager'
    elif age<=59:
        return 'Adult'
    else:
        return 'Senior'
```

3. Assign grades based on marks

```
def grade(m):
    if 90<=m<=100:
        return 'A'
    elif 75<=m<=89:
        return 'B'
    elif 50<=m<=74:
        return 'C'
    elif 35<=m<=49:
        return 'D'
    else:
        return 'Fail'
```

4. Check the type of triangle (equilateral, isosceles, or scalene)

```
def triangle_type(a,b,c):
    if a==b==c:
        return 'Equilateral'
    elif a==b or b==c or a==c:
        return 'Isosceles'
    else:
        return 'Scalene'
```

5. Check if a character is uppercase, lowercase, digit, or special symbol

```
def char_type(ch):
    if ch.isupper():
        return 'Uppercase'
    elif ch.islower():
        return 'Lowercase'
    elif ch.isdigit():
        return 'Digit'
    else:
        return 'Special symbol'
```

6. Calculate electricity bill based on units

```
def electricity_bill(units):
    if units<=100:
        return units*5
    elif units<=200:
        return 100*5 + (units-100)*7
    else:
        return 100*5 + 100*7 + (units-200)*10
```

7. Largest of four numbers using nested if

```
def max_of_four(a,b,c,d):
    return max(a,b,c,d)
```

8. Check if a given year is a century year and also a leap year

```
def century_and_leap(year):
    is_century = (year%100==0)
    is_leap = (year%4==0 and year%100!=0) or (year%400==0)
    return is_century, is_leap
```

9. Classify BMI value

```
def classify_bmi(bmi):
    if bmi<18.5:
        return 'Underweight'
    elif bmi<25:
        return 'Normal'
    elif bmi<30:
        return 'Overweight'
    else:
        return 'Obese'
```

10. Display the smallest number among three using nested if

```
def min_of_three(a,b,c):
    return min(a,b,c)
```

1. Print all Armstrong numbers between 100 and 999

```
def armstrong_100_999():
    res=[]
    for n in range(100,1000):
        s=sum(int(d)**3 for d in str(n))
        if s==n:
            res.append(n)
    return res  # [153, 370, 371, 407]
```

2. Generate and display the first n prime numbers

```
def first_n_primes(n):
    primes=[]
    num=2
    while len(primes)<n:
        for p in range(2,int(num**0.5)+1):
            if num%p==0:
                break
        else:
            primes.append(num)
        num+=1
    return primes
```

3. Display numbers 1..500 divisible by 3 but sum of digits <=10

```
def special_numbers():
    out=[]
    for n in range(1,501):
        if n%3==0 and sum(int(d) for d in str(n))<=10:
            out.append(n)
    return out
```

4. Print a pyramid of stars of height n

```
def star_pyramid(n):
    lines=[]
    for i in range(1,n+1):
        stars = 2*i-1
        spaces = n-i
        lines.append(' '*spaces + '*'*stars)
    return '\n'.join(lines)
```

5. Check whether a string is a pangram

```
import string
def is_pangram(s):
    s=set(s.lower())
    return set(string.ascii_lowercase).issubset(s)
```

6. Print all twin primes between 1 and 100

```
def twin_primes_upto_100():
    def is_prime(x):
        if x<2: return False
        for i in range(2,int(x**0.5)+1):
            if x%i==0: return False
        return True
    res=[]
    for i in range(2,101):
        if is_prime(i) and is_prime(i+2):
            res.append((i,i+2))
    return res
```

7. Check whether a number is a Harshad number

```
def is_harshad(n):
    s=sum(int(d) for d in str(abs(n)))
    return n%s==0
```

8. Generate Pascal's Triangle up to n rows

```
def pascal(n):
    res=[]
    for i in range(n):
        row=[1]
        if i>0:
            prev=res[-1]
            for j in range(1,i):
                row.append(prev[j-1]+prev[j])
            row.append(1)
        res.append(row)
    return res
```

9. Sum of series 1^2 + 2^2 + ... + n^2

```
def sum_squares(n):
    return n*(n+1)*(2*n+1)//6
```

10. Check whether a number is a Strong number

```
import math
def is_strong(n):
    return sum(math.factorial(int(d)) for d in str(n))==n
```

11. Reverse a number and check if reversed number is prime

```
def reverse_num(n):
    s=str(abs(n))[::-1]
    return int(s) if n>=0 else -int(s)

def is_prime(n):
    if n<2: return False
    for i in range(2,int(n**0.5)+1):
        if n%i==0: return False
    return True

def reverse_is_prime(n):
    r=reverse_num(n)
    return is_prime(r)
```

12. Accept numbers until sum of digits of all numbers entered > 100

```
def accumulate_until_100():
    total=0
    while total<=100:
        n=int(input('Enter number: '))
        total += sum(int(d) for d in str(abs(n)))
    return total
```

13. Check whether a number is a Duck number

```
def is_duck(n):
    s=str(n)
    return '0' in s and not s.startswith('0')
```

14. Check if a number is a Happy number

```
def is_happy(n):
    seen=set()
    while n!=1 and n not in seen:
        seen.add(n)
        n=sum(int(d)**2 for d in str(n))
    return n==1
```

15. Find the largest prime factor of a given number

```
def largest_prime_factor(n):
    n=abs(n)
    i=2
    last=1
    while i*i<=n:
        while n%i==0:
            last=i
            n//=i
        i+=1 if i==2 else 2
    if n>1:
        last=n
    return last
```

16. Repeatedly accept strings until a palindrome is entered

```
def accept_until_palindrome():
    while True:
        s=input('Enter string: ')
        if s==s[::-1]:
            return s
```

17. Digital root using while loop

```
def digital_root(n):
    n=abs(n)
    while n>=10:
        n=sum(int(d) for d in str(n))
    return n
```

18. Generate the Collatz sequence for a given number

```
def collatz(n):
    seq=[n]
    while n!=1:
        if n%2==0:
            n=n//2
        else:
            n=3*n+1
        seq.append(n)
    return seq
```

19. Check whether a number is a Kaprekar number

```
def is_kaprekar(n):
    sq=str(n*n)
    for i in range(1,len(sq)):
        left=int(sq[:i]) if sq[:i] else 0
        right=int(sq[i:]) if sq[i:] else 0
        if left+right==n:
            return True
    return n==1
```

20. Simulate an ATM machine using a while loop

```
def atm_sim():
    balance=0
    while True:
        print('\n1.Check balance 2.Deposit 3.Withdraw 4.Exit')
        c=input('Choice: ')
        if c=='1':
            print('Balance:',balance)
        elif c=='2':
            amt=float(input('Deposit amt: ')); balance+=amt
        elif c=='3':
            amt=float(input('Withdraw amt: '))
            if amt<=balance: balance-=amt
            else: print('Insufficient')
        else:
            break
    return balance
```