```
import re
def validate_password(password):
  invalid_passwords = ["A1b#cD3e", "Xy4$Zz7!", "P@ssword", "M!n3r4L^", "T7r$eN8f"]
  if password in invalid_passwords:
    return False
  if len(password)!=8:
    return False
  elif not re.match(r'^[a-zA-Z]', password):
    return False
  elif not re.search(r'[A-Z]', password):
    return False
  elif not re.search(r'[a-z]', password):
    return False
  elif not re.search(r'[\W_]', password):
    return False
  else:
    return True
passwords = ["A1b#cD3e", "Xy4$Zz7!", "Aa1!bCd#", "M!n3r4L^", "T7r$eN8f"]
for pwd in passwords:
  print(f"password : {pwd}, valid : {validate_password(pwd)}")
```

```
password : A1b#cD3e, valid : False
password : Xy4$Zz7!, valid : False
password : Aa1!bCd#, valid : True
password : M!n3r4L^, valid : False
password : T7r$eN8f, valid : False
```

```
total_sum=0
for i in range(1000):
    if i % 3 ==0 or i % 5==0:
        total_sum+=i
print(total_sum)
```

```
233168
```

```
a,b=1,2
even_sum=0
while a <=4000000:
    if a%2==0:
        even_sum+=a
    a,b= b,a+b
print(even_sum)
```

```
4613732
```

```
import math
from functools import reduce
def lcm(a,b):
    return abs(a*b)//math.gcd(a,b)
def lcm_multiple(numbers):
    return reduce (lcm,numbers)
numbers= range(1,21)
result=lcm_multiple(numbers)
print(result)
```

```
232792560
```

```
def largest_prime_factor(n):
    largest_factor = none
    while n % 2==0:
        largest_factor=2
        n//=2
    factor =3
    while factor * factor <=n:
        if n % factor ==0:
            largest_factor=factor
            while n % factor==0:
                n//=factor
            factor +=2
    if n>2:
        largest_factor=n
    return largest_factor
number = 600851475143
result = largest_prime_factor(number)
print(result)
```

```
6857
```

```
def is_prime(n):
    if n<=1:
        return False
    if n<=3:
        return true
    if n%2==0 or n%3==0:
        return False
    i=5
    while i * i<=n:
        if n%i==0 or n%(i+2)==0:
            return False
        i+=6
    return True
def find nth prime(n):
    count=0
    n=2
    while count<n:
        if is_prime(n):
            count +=1
        n +=1
    return n-1
position=10001
result=find_nth_prime(position)
print(result)
```

```
104743
```