Find all natural numbers belonging to the segment [35,000,000; 40,000,000], which have exactly five different odd divisors (the number of even divisors can be any). In the answer, list the numbers found in ascending order

324

1, 2, 3, 4, 6, 9, 12, 18, 27, 36, 54, 81, 108, 162, 324

81=3^4

162=2*3^4

324=4*3^4

625=5^4

648=8*3^4

35819648
38950081
39037448
39337984

2^k*p^4
1
p
p^2
p^3
p^4

2^k*p^2*q^2
1
p
p^2
q
q^2
q*p
q^2*p^2

2^k*p*q^2
1
p
q
q^2
pq
pq^2

2^k*p*q
1
p
q
pq

2^k*p*q*t
1
p
q
t
pq
pt
qt
pqt

```
ms=[]
x=35000000
while x<=40000000:
  k=x
  firstdivisor=0
  flag=1
  flagbreak=0
  n=2
  counter=0
  v=k**0.5
  while n<=v:
    if k%n==0:
      if n%2!=0 and (n==firstdivisor or firstdivisor==0):
        if flag==1:
          firstdivisor=n
        if firstdivisor==n:
          flag=2
          counter+=1

#print("counter=",counter,"k=",k,"n=",n,"firstdivisor=",firstdivisor,"flag=",flag)
      elif n%2!=0:
        flagbreak=1
        break
      k=k//n
      v=k**0.5
      n-=1
    n+=1
  #print(counter,k,flagbreak)
  if k%2!=0 and k==firstdivisor:
    counter+=1
  elif k%2!=0:
    flagbreak=1
  #print(counter)
  if counter==4 and flagbreak==0:
    ms.append(x)
  x+=1
#print(flagbreak)
print(ms)
```

22 min

```
void betterOddDivisor(int number, int secondNum) {
  int temp;
  int counter;
  int flag;
  int flagBreak;
  for (int k = number; k <= secondNum; k++) {
     temp = k;
     counter = 0;
     double root = sqrt(temp);
     flag = 0;
     flagBreak = 0;
     for (int i = 2; i <= root; i++) {
        if (temp % i == 0) {
           if (i % 2 != 0) {
              if (flag == 0) {
                 flag = i;
                 counter++;     // starts counting 1st odd divisor instance
              }
              else if (i != flag) {
                 flagBreak = 1;
                 break;         // this stops the  'for' loop, not just the 'ifs' (encountered a different odd
divisor)
              }
              else if (i == flag) {
                 counter++;       // encountered another odd divisor similar to 1st instance
              }
           }
           temp = temp / i;
           root = sqrt(temp);
           i--;
        }
     }
     if (flagBreak == 0) {
        if (temp % 2 != 0) {   // adds last encounter of the divisor for the '5th' count
           if (temp == flag) {
              counter++;
           }
           else if (temp != flag) {
              flagBreak = 1;
           }
        }
     }
     if (flagBreak == 0) {
        if (counter == 4) {
           std::cout << k << std::endl;
        }
     }
  }
}
```

22 sec

```
ms=[]
x=35000000
while x<40000000:
  k=x
  n=1
  f=0
  while n<=k:
    if n%2!=0:
      if k%n==0:
        f+=1
    n+=1
  if f==5:
    ms.append(x)
  x+=1
print(ms)
```