```
gap :=proc(a, b)
  return abs(a - b) - 1;
 end proc
```

```
gap := proc(a, b) return abs(a - b) - 1 end proc
```

(1)

```
HostsNdivisors :=proc(N)
 local i, j, g, d, L, s, t;
  local m, p, q, P, Q, np, nq;
  m := floor((N-1)/2);
  L := evalf(sqrt(N));
  P := Array( );
  Q := Array( );
  s := 1; t := 1;

   for i from 3 to m do
    d := gcd(i, N);
    if 1 < d and d < L then
    P(s) := i; s++;
   elif d > L then   Q(t) := i; t++; fi;
    od;
    np := s - 1;
    nq := t - 1;
   for i from 1 to  np do printf("%3d,", P(i));od;

   printf("\n");
      for i from 1 to  nq do printf("%3d,", Q(i)); od;

    printf("\n gaps: \n");


   for i from 1 to np do
     for j from 1 to nq do
      p := P(i);
      q := Q(j);
      g := gap(p, q);
      printf("%4d,", g);
     od;
     printf("\n");
   od;


 end proc
```

```
HostsNdivisors := proc(N)
   local i, j, g, d, L, s, t, m, p, q, P, Q, np, nq;
   m := floor(1/2*N - 1/2);
   L := evalf(sqrt(N));
```

(2)

```
    P := Array( );
    Q := Array( );
    s := 1;
    t := 1;
    for i from 3 to m do
        d := gcd(i, N);
        if 1 < d and d < L then
            P(s) := i; `++`(s)
        elif L < d then
            Q(t) := i; `++`(t)
        end if
    end do;
    np := s - 1;
    nq := t - 1;
    for i to np do printf("%3d,", P(i)) end do;
    printf("\n");
    for i to nq do printf("%3d,", Q(i)) end do;
    printf("\n gaps: \n");
    for i to np do
        for j to nq do
            p := P(i); q := Q(j); g := gap(p, q); printf("%4d,", g)
        end do;
        printf("\n")
    end do
end proc
```

```
HostOfpq :=proc(p, q)
 local α, s, t, g, r, S, T;
 local i, j;
```

$$S := \frac{q-1}{2};$$

$$T := \frac{p-1}{2};$$

$$\alpha := \mathrm{floor}\left(\frac{q}{p}\right);$$

$$r := q - \alpha \cdot p;$$

```
  for s from 1 to S do
   for t from 1 to T do
```

$$g := \mathrm{abs}\left(\left(t \cdot \alpha - s\right) \cdot p + t \cdot r\right) - 1;$$

```
    printf("%4d,", g);
```

```
    od;
    printf("\n");
  od;
end proc
```

(3)

```
HostOfpq := proc(p, q)
   local alpha, s, t, g, r, S, T, i, j;
   S := 1/2*q - 1/2;
   T := 1/2*p - 1/2;
   alpha := floor(q/p);
   r := q - alpha*p;
   for s to S do
      for t to T do
         g := abs((t*alpha - s)*p + t*r) - 1; printf("%4d,", g)
      end do;
      printf("\n")
   end do
end proc
HostsNdivisors(55)
  5, 10, 15, 20, 25,
 11, 22,
 gaps:
   5,  16,
   0,  11,
   3,   6,
   8,   1,
  13,   2,


HostOfpq(5, 11)
   5,  16,
   0,  11,
   3,   6,
   8,   1,
  13,   2,
```