Sifting for Primes in Different Languages

Here are two listings of code for running the sieve of Eratosthenes to generate all primes up to the bound represented by the given argument.

gp

```
\\ sift(n) = list of all primes < = n
sift(n) =
{
local(v,w,j,k,l,p,r);
v = vector(n,j,j);
l = listcreate(n);
r=1;
k=1;
while(r<n,k=r;\
  while(v[k]==1,k++;if(k>n,break(2),));\
  r=k;\
  p=v[r];\
  listput(l,p);\
  for(j=1,floor(n/p),v[j*p]=1));
w = vector(length(l),j,l[j]);
return(w);
}
```

sage

```
# sift(n) = list of all primes < = n
def sift(n):
  v = range(0,n+1);
  l = [];
  r=1;
  k=1;
  c=0;
  while (r<n and c < n+1):
    c = c + 1;
    k=r;
    while v[k]==1:
      k = k + 1;
      if k > n:
        return(l);
    r = k;
    p = v[r];
    l.append(p);
    s = floor(n/p);
    for j in range(1,s+1):
      t = j*p;
      v[t] = 1;
  return(l);
```