// The output are the standard generators of cyclic codes of given n and q.

function baseq\_rep(n,q,irrNum)

listPowers:=[];

for i in [1..irrNum] do

r := n mod q;

Append(~listPowers, r);

n := n div q;

end for;

return listPowers;

end function;

// this function takes a list of powers and the factorization list of lists

// the output is the corresponding polynomial

function vectorToPoly(vecPower,vecFac)

poly:=1;

for i in [1..#vecPower] do

poly:=poly\*vecFac[i]^vecPower[i];

end for;

return poly;

end function;

Z4:=IntegerRing(4);

P4<x>:=PolynomialRing(Z4);

Fac:=CyclotomicFactors(Z4, n);

power:=1;

file:="/home/lu1/AllFreeCyclicZ\_4/leeOutput/CyclicCodesLen" cat IntegerToString(n) cat ".txt";

PrintFile(file, "&&&");

for num in [0..(((power+1)^#Fac)-1)] do

startTime:=Cputime();

listPower:=baseq\_rep(num,power+1,#Fac);

generator:=vectorToPoly(listPower,Fac);

C:=CyclicCode(n,generator);

if (HasLinearGrayMapImage(C)) then

PrintFile(file, "Linear Gray Map Image");

else

PrintFile(file, "Nonlinear Gray Map Image");

end if;

PrintFile(file, "&");

PrintFile(file, n-Degree(generator));

PrintFile(file, "&");

PrintFile(file, MinimumLeeWeight(C));

PrintFile(file, "&");

PrintFile(file, generator);

PrintFile(file, "&");

PrintFile(file, Cputime()-startTime);

PrintFile(file, "&&&");

end for;