Finite Rings and Fields Solutions

Computer Algebra for Cryptography (B-KUL-H0E74A)

Exercise 1.

- ```
  Z := Integers();
  N := 105;
  ZN := Integers(N);
  ```

- ```
  for i i:= 1 to 10 do
    a := Random(ZN);
    a;
    a^-1;
  end for;
  ```

  Trying to invert 10 random elements will almost always result in an error since more than half of the elements of $\mathbb{Z}_{105}$ are simply not invertible.

- ```
  a := Random(ZN);
  g, x, _ := XGCD(Z ! a,N);
  if g eq 1 then
     ZN ! x, (ZN ! a)^-1;
  else
     "The chosen element is not invertible.";
  end if;
  ```

- One can check that $\varphi(105) = 48$ hence exponentiation to the power 47 should result in the inverse of an element over $\mathbb{Z}_{105}$ (in case the element is invertible).

  ```
  a := Random(ZN);
  if GCD(Z!a,N) ne 1 then
    print "The chosen element is not invertible.";
  else
    print a^(EulerPhi(N)-1);
  end if;
  ```

  To show that all these methods yield the same result, you can use a loop such as the following:
  ```

```
check := true;
for a in ZN do
  if GCD(Z!a,N) eq 1 then
    ainv1 := a^-1;
    _, x, _ := XGCD(Z!a,N);
    ainv2 := (ZN ! x);
    ainv3 := a^(EulerPhi(N)-1);
    check := check and (ainv1 eq ainv2) and (ainv2 eq ainv3);
  end if;
end for;
check;
```

Exercise 2.

```
p := 31;
Fp := GF(31);
&and[a^p eq a : a in Fp];
```

Exercise 3.

- ```
  p := 31;
  Fp := GF(p);
  R<x> := PolynomialRing(Fp);
  ```
- ```
  function GenRandomIrreduciblePol(K, n)
    R<x> := PolynomialRing(K);
    repeat
      pol := R ! ([Random(K) : i in [1..n]] cat [1]);
     until IsIrreducible(pol);
    return pol;
  end function;
  ```

Exercise 4.

- ```
  p := 31;
  Fp := GF(p);
  f := RandomIrreduciblePolynomial(Fp,3);
  ```
- ```
  Fpn<w> := ext<Fp | f>;
  Evaluate(f,w);
  ```
- ```
  a := Random(Fpn);
  print a;
  ```
- ```
  SetPowerPrinting(Fpn, false);
  ```

```
pol := PolynomialRing(Fp) ! Eltseq(a);
_, inv, _ := XGCD(pol,f);
(Fpn ! inv) eq a^-1;
```

- ```
  for i := 1 to 10 do
    print RandomIrreduciblePolynomial(Fp,3);
  end for;
  ```

Exercise 5.

- ```
  p := 31;
  Fp := GF(p);
  R<x> := PolynomialRing(Fp);
  n := 3;
  f := RandomIrreduciblePolynomial(Fp,n);
  repeat g := RandomIrreduciblePolynomial(Fp,n); until f ne g;
  ```

- ```
  Fpn1<w1> := ext<Fp | f>;
  Roots(f,Fpn1);
  Roots(g,Fpn1);
  ```

- ```
  Fpn2<w2> := ext<Fp | g>;
  Roots(f,Fpn2);
  Roots(g,Fpn2);
  ```