Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

“Спеціальні розділи обчислювальної математики”

Комп’ютерний практикум

Робота №3. Реалізація операцій у скінченних полях характеристики 2 (поліноміальний базис)

Виконала
студентка гр. ФБ-11 Данькова Єлізавета

Мета роботи

Одержання практичних навичок програмної реалізації обчислень у полі Галуа характеристики 2 в поліноміальному базисі; ознайомлення з прийомами ефективної реалізації критичних по часу ділянок програмного коду та методами оцінки їх ефективності.

ХІД РОБОТИ:

Додавання

```
def addition(poly_1, poly_2):
    poly_1 = poly_1.zfill(173)
    poly_2 = poly_2.zfill(173)
    c = ['0']
    for i in range(173):
        c.append(str(((int(poly_1[i]) if i < len(poly_1) else 0) +
(int(poly_2[i]) if i < len(poly_2) else 0)) % 2))
    result = ''.join(c)
    return result.lstrip('0') if len(result) <= 173 else result[(len(result)
- 173):]
```

Множення

```
def multiply_polynomials(poly_1, poly_2, mod_poly):
    deg_poly1 = len(poly_1) - 1
    deg_poly2 = len(poly_2) - 1
    result = [0] * (deg_poly1 + deg_poly2 + 1)
    for i in range(deg_poly1 + 1):
        for j in range(deg_poly2 + 1):
            result[i + j] ^= int(poly_1[i]) & int(poly_2[j])
    while result and result[0] == 0:
        result.pop(0)
    result = ''.join(map(str, result))
    result = binary_division(result, int(mod_poly, 2))
    return result
```

Квадрат

```
def square(poly_1, mod_poly):
    sq_ = multiply_polynomials(poly_1, poly_1, mod_poly)
    return sq_
```

Обернений

```
def reverse(poly_1):
    power_reverse = 2 ** 173 - 2
    power_reverse = bin(power_reverse)[2:]
    power_reverse = str(power_reverse)
    return power_poly(poly_1, power_reverse, mod)
```

Степінь

```
def power_poly(poly_1, poly_3, mod_poly):
    result = '1'
    binary_exponent = ''.join(map(str, poly_3))
    for bit in binary_exponent[::-1]:
        if bit == '1':
            result = multiply_polynomials(result, poly_1, mod_poly)
        poly_1 = square(poly_1, mod_poly)
    return result
```