

1. Алгоритм работы DES (текст + блок-схема/примерный код)

Общая идея

DES — симметричный блочный шифр длиной блока 64 бита, использующий ключ длиной 56 бит. Он выполняет 16 раундов сложных перестановок и подстановок.

Основные шаги алгоритма:

Разделение входных данных:

Входной блок (64 бита) делится на две части: левую (L) и правую (R) по 32 бита.

Начальная перестановка (Initial Permutation, IP):

Перестановка битов входного блока по фиксированной таблице.

Раунды (16 раундов):

Для каждого раунда:

Создается расширенная версия R (расширение с 32 до 48 бит).

К расширенной версии применяется раундовый ключ (48 бит).

Полученное значение пропускается через функцию F.

Результат XOR с L.

После этого L и R меняются местами (кроме последнего раунда).

Обратная перестановка (Final Permutation, FP):

После 16 раундов блоки объединяются и применяют обратную перестановку.

Примерный псевдокод:

Вход: 64-битный блок data, 56-битный ключ key

Выход: зашифрованный 64-битный блок

```
def des_encrypt(data, key):
```

```
    # 1. Начальная перестановка
```

```
    data = initial_permutation(data)
```

```

# 2. Генерация 16 раундовых ключей
round_keys = generate_round_keys(key)

# 3. Разделение
L, R = split_into_left_right(data)

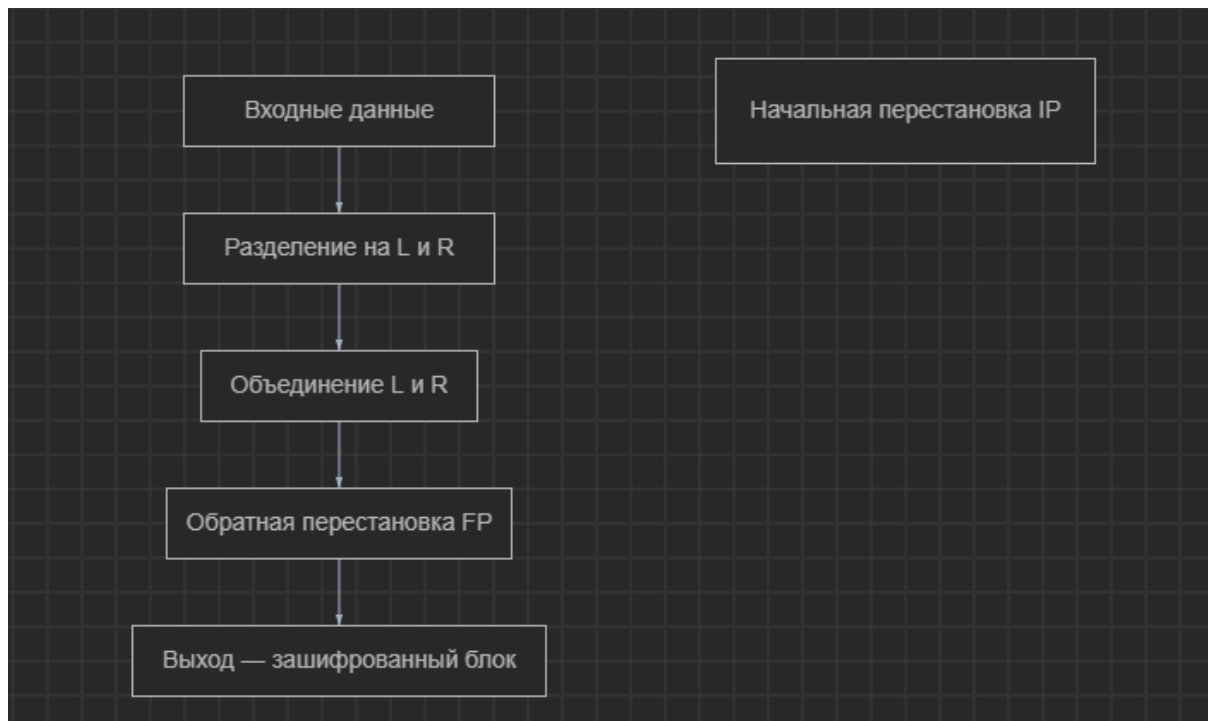
# 4. 16 раундов
for i in range(16):
    temp = R
    R = L ^ F(R, round_keys[i])
    L = temp

# 5. Объединение и обратная перестановка
pre_output = combine(L, R)
cipher_text = final_permutation(pre_output)

return cipher_text

```

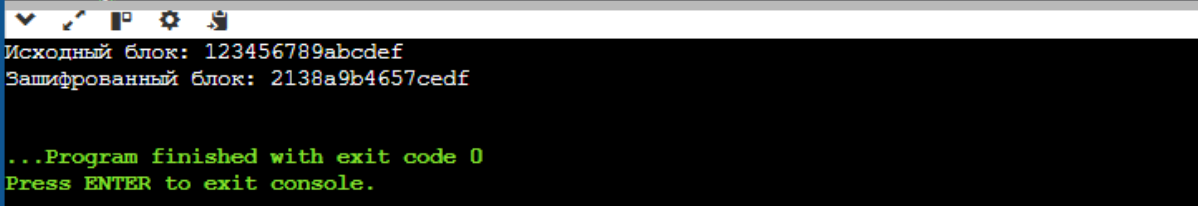
Блок-схема :



2. Реализация (код на C++)

Ниже приведён пример реализации DES.

```
162     for (int i = 0; i < 16; ++i) {
163         uint32_t R_next = L ^ feistel(R, subkeys[i]);
164         L = R;
165         R = R_next;
166     }
167
168     uint64_t pre_output = ((uint64_t)R << 32) | L;
169     uint64_t cipher = permute(pre_output, FP, 64);
170     return cipher;
171 }
172
173 int main() {
174     uint64_t plaintext = 0x0123456789ABCDEFULL;
175     uint64_t key = 0x1334577998BCDFF1ULL;
176
177     uint64_t ciphertext = des_encrypt_block(plaintext, key);
178
179     std::cout << "Исходный блок: " << std::hex << plaintext << std::endl;
180     std::cout << "Зашифрованный блок: " << std::hex << ciphertext << std::endl;
181
182     return 0;
183 }
```



```
Исходный блок: 123456789abcdef
Зашифрованный блок: 2138a9b4657cedf

...Program finished with exit code 0
Press ENTER to exit console.
```

3 Итог

Алгоритм DES включает начальную перестановку, 16 раундов с функцией F, обмен L и R, и обратную перестановку.

Реализация может быть выполнена на любом языке, с использованием таблиц перестановок и S-блоков.

Тестирование проводится на стандартных тестовых данных, результат которых сравнивается с эталонными.