EXPERIMENT-6

AIM: To implement DES sub key generation.

THEORY:

The DES (Data Encryption Standard) subkey generation process derives sixteen 48-bit round keys from an initial 64-bit key to be used in each round of encryption and decryption. The 64-bit key is first permuted using the PC-1 table to produce a 56-bit key, which is then split into two halves (C and D). These halves undergo a series of left circular shifts based on a predefined shift schedule. After each shift, both halves are combined and permuted again using the PC-2 table to produce a unique 48-bit subkey. This process ensures that each round of DES uses a different key, enhancing diffusion and overall security.

CODE:

```
// To implement DES sub key generation By Aaditya Bhatia 23/CS/004
#include <iostream>
#include <vector>
using namespace std;

int PC1[56] = {
   57, 49, 41, 33, 25, 17, 9,
   1, 58, 50, 42, 34, 26, 18,
   10, 2, 59, 51, 43, 35, 27,
   19, 11, 3, 60, 52, 44, 36,
   63, 55, 47, 39, 31, 23, 15,
   7, 62, 54, 46, 38, 30, 22,
   14, 6, 61, 53, 45, 37, 29,
   21, 13, 5, 28, 20, 12, 4
};

int PC2[48] = {
   14, 17, 11, 24, 1, 5,
   3, 28, 15, 6, 21, 10,
```

```
    23, 19, 12, 4, 26, 8,
    16, 7, 27, 20, 13, 2,
    41, 52, 31, 37, 47, 55,
    30, 40, 51, 45, 33, 48,
    44, 49, 39, 56, 34, 53,
    46, 42, 50, 36, 29, 32
};

int SHIFTS[16] = {
    1, 1, 2, 2,
    2, 2, 2, 2,
    1, 2, 2, 2,
    2, 2, 2, 1
};

string permute(string input, int *table, int n) {
    string output = "";
    for (int i = 0; i < n; i++)
        output += input[table[i] - 1];
    return output;
}

string leftRotate(const string &key, int shifts) {
    return key.substr(shifts) + key.substr(0, shifts);
}

int main() {
    string key64;
    cout << "Enter 64-bit key (as binary string): ";
    cin >> key64;
    if (key64.size() != 64)
    {
        cerr << "Key must be 64 bits" << endl;
        return 1;
    }
    string key56 = permute(key64, PC1, 56);
    string C = key56.substr(0, 28);
    string D = key56.substr(28, 28);
    cout << "Round Subkeys:" << endl;
```

```
    for (int round = 0; round < 16; round++)
    {
        C = leftRotate(C, SHIFTS[round]);
        D = leftRotate(D, SHIFTS[round]);
        string CD = C + D;
        string subkey = permute(CD, PC2, 48);
        cout << "Round " << round + 1 << ": " << subkey << endl;
    }
    return 0;
}
```

OUTPUT:

```
aadi@Joshua:~/Projects/LABS2025/INS Lab$ g++ 6.cpp -o 6 && ./6
Enter 64-bit key (as binary string): 1010110011101011001110101100111010110011101011001111010110011111 0
Round Subkeys:
Round 1: 110110100110111100111001001000011111110111101111
Round 2: 011011111110100011100111011111100111111100011001
Round 3: 101110111110010110101110101110111011000101111010
Round 4: 111110000000011110111011110010111111011001000010
Round 5: 111101011001101000111101111101000010111001111110
Round 6: 100001111011101011010111011111101100110101011110
Round 7: 001111100111111011110110000101011111011011111011
Round 8: 111111100111010101101000001111111011110001100101
Round 9: 010001111011100111110111011110100101001010111111
Round 10: 101111111110010011110011110101110111100110101011
Round 11: 111111110100011110101010101001100011101101111001
Round 12: 111110101001001110011101111100111011101101110110
Round 13: 000111011001101001011111011101011000111110111010
Round 14: 001001110111100011111110010111010011110001011111
Round 15: 101111100110110111100100011011111111000011111100
Round 16: 010101111001110101010111110110101110111010101011
aadi@Joshua:~/Projects/LABS2025/INS Lab$
```

LEARNING OUTCOME:

To understand and implement the DES subkey generation process using permutation and left-shift operations for secure key scheduling.