Program:

```
from cryptography.fernet import Fernet
import os

def generate_key(key_file='secret.key'):
    """Generate a new encryption key and save it to a file."""
    key= Fernet.generate_key()
    with open(key_file, 'wb') as keyfile:
        keyfile.write(key)
    print(f"Key saved to {key_file}")

def load_key(key_file='secret.key'):
    """Load the encryption key from a file."""
    if not os.path.exists(key_file):
        raise FileNotFoundError(f"Key file {key_file} not found. Generate a key first.")
    with open(key_file, 'rb') as keyfile:
        return keyfile.read()

def encrypt_message(message, key_file='secret.key'):
    """Encrypt a message using the encryption key."""
    key= load_key(key_file)
    fernet = Fernet(key)
    encrypted_message = fernet.encrypt(message.encode())
    return encrypted_message

def decrypt_message(encrypted_message, key_file='secret.key'):
    """Decrypt a message using the encryption key."""
    key= load_key(key_file)
    fernet = Fernet(key)
    decrypted_message = fernet.decrypt(encrypted_message).decode()

    return decrypted_message
```

```
def encrypt_file(file_path, key_file='secret.key'): """Encrypt the
    contents of a file."""
    key= load_key(key_file) fernet = Fernet(key)
    with open(file_path, 'rb') as file:
        original_data = file.read() encrypted_data =
    fernet.encrypt(original_data) with open(file_path, 'wb') as
    file:
        file.write(encrypted_data)
print(f"File {file_path} encrypted successfully.")

def decrypt_file(file_path, key_file='secret.key'): """Decrypt the
    contents of a file."""
    key= load_key(key_file) fernet = Fernet(key)
    with open(file_path, 'rb') as file:
        encrypted_data = file.read() decrypted_data =
    fernet.decrypt(encrypted_data) with open(file_path, 'wb') as
    file:
        file.write(decrypted_data)
print(f"File {file_path} decrypted successfully.")

if   name   == "_main_"·
print("Welcome to the Encryption Tool")
print("1. Generate Key")
print("2. Encrypt Message")
print("3. Decrypt Message")
print("4. Encrypt File") print("S. Decrypt File")
choice= input("Enter your choice: ")

if choice== '1': generate_key()
elif choice== '2':
    message= input("Enter the message to encrypt: ") encrypted=
    encrypt_message(message) print(f"Encrypted Message: {encrypted}")
elif choice== '3':
```

```
    encrypted_message = input("Enter the encrypted message: ").encode()
    try:
decrypted= decrypt_message(encrypted_message) print(f"Decrypted Message:
{decrypted}")
    except Exception as e:
print("Decryption failed:", e)
 elif choice== '4':
    file_path = input("Enter the file path to encrypt: ")
    try:
        encrypt_file(file_path)
    except Exception as e:
print("Encryption failed:", e)
 elif choice== '5':
    file_path = input("Enter the file path to decrypt: ")
    try:
        decrypt_file(file_path) except Exception
    as e:
print("Decryption failed:", e)
 else:
    print("Invalid choice. Exiting.")
```

Output:

```
Welcome to the Encryption Tool

1.    Generate Key
2.    Encrypt Message
3.    Decrypt Message
4.    Encrypt File
5.    Decrypt File

Enter your choice:

Error: Command failed: timeout 7 python3 main.py
```