```
.data
filename: .asciiz "SO_BDH.TXT"
buffer: .space 64
newline: .asciiz "\n"
decimal_msg: .asciiz "Ket qua he 10: "
hex_msg: .asciiz "Ket qua he 16: "
binary_msg: .asciiz "Ket qua he 2: "

.text
main:
    # Thiết lập seed dựa trên thời gian hiện tại
    li $v0, 30       # syscall 30: lấy thời gian hiện tại
    syscall
    move $a0, $v0    # di chuyển thời gian vào $a0
    li $v0, 40       # syscall 40: set seed
    syscall

    # Tạo số ngẫu nhiên
    li $v0, 42       # syscall 42: random number
    syscall
    andi $a0, $v0, 0xFFFF # Giới hạn số trong khoảng 0 đến 65535

    # Lưu $a0 vào một vị trí khác để sử dụng sau này
    move $t0, $a0

    # Mở tập tin để ghi
    li $v0, 13       # syscall 13: mở tập tin
    la $a0, filename # đường dẫn tập tin
    li $a1, 577      # quyền ghi
    li $a2, 0
    syscall
    move $s6, $v0    # lưu file descriptor vào $s6

    # Chuyển đổi số sang hệ 10 và ghi vào tập tin
    jal convert_decimal
    la $a1, decimal_msg
    jal write_file

    # Chuyển đổi số sang hệ 16 và ghi vào tập tin
    move $a0, $t0
    jal convert_hex
    la $a1, hex_msg
    jal write_file

    # Chuyển đổi số sang hệ 2 và ghi vào tập tin
    move $a0, $t0
```

```
    jal convert_binary
    la $a1, binary_msg
    jal write_file

    # Đóng tập tin
    li $v0, 16       # syscall 16: đóng tập tin
    move $a0, $s6
    syscall

    # Kết thúc chương trình
    li $v0, 10       # syscall 10: thoát
    syscall

# Hàm chuyển đổi số sang hệ 10
# Hàm chuyển đổi số sang hệ 10
convert_decimal:
    li $t1, 10         # Đặt cơ số là 10
    li $t2, 0          # $t2 sẽ chứa số lượng chữ số
    li $t3, 0          # Biến tạm để lưu kết quả lấy dư
    li $t4, '0'        # Ký tự '0' để chuyển đổi số sang ký tự

    # Đảo ngược số để dễ dàng xử lý từ phải sang trái
reverse_loop:
    beq $a0, $zero, reverse_done  # Nếu số đã hết chữ số, kết thúc vòng lặp
    div $a0, $t1                  # Chia số cho 10
    mfhi $t3                      # Lấy phần dư
    addi $t3, $t3, '0'            # Chuyển số thành ký tự
    sb $t3, buffer($t2)           # Lưu ký tự vào buffer
    mflo $a0                      # Lấy phần nguyên của kết quả chia
    addi $t2, $t2, 1              # Tăng số lượng chữ số
    j reverse_loop                # Lặp lại

reverse_done:
    # Đảo chuỗi để đúng thứ tự
    li $t5, 0          # $t5 là chỉ số của chuỗi đảo ngược
reverse_string_loop:
    blt $t5, $t2, continue_reversing  # Nếu chưa xử lý hết chuỗi
    j reverse_string_done             # Hoàn tất đảo chuỗi

continue_reversing:
    sub $t6, $t2, $t5                # Tính vị trí đối xứng
    subi $t6, $t6, 1                 # Điều chỉnh chỉ số
    lb $t7, buffer($t5)              # Lấy ký tự tại vị trí hiện tại
    lb $t8, buffer($t6)              # Lấy ký tự tại vị trí đối xứng
    sb $t7, buffer($t6)              # Hoán đổi các ký tự
    sb $t8, buffer($t5)
```

```
    addi $t5, $t5, 1                # Tăng chỉ số
    j reverse_string_loop           # Lặp lại

reverse_string_done:
    # Thêm ký tự kết thúc chuỗi
    sb $zero, buffer($t2)

    # Trở về chương trình chính
    jr $ra


# Hàm chuyển đổi số sang hệ 16
# Hàm chuyển đổi số sang hệ 16
convert_hex:
    li $t1, 16        # Đặt cơ số là 16
    li $t2, 0         # $t2 sẽ chứa số lượng chữ số
    li $t3, 0         # Biến tạm để lưu kết quả lấy dư

    # Đảo ngược số để dễ dàng xử lý từ phải sang trái
reverse_hex_loop:
    beq $a0, $zero, reverse_hex_done  # Nếu số đã hết chữ số, kết thúc vòng
lặp
    div $a0, $t1                    # Chia số cho 16
    mfhi $t3                        # Lấy phần dư
    blt $t3, 10, convert_digit      # Chuyển đổi số sang ký tự '0' đến '9'
    addi $t3, $t3, 55               # Chuyển đổi số thành ký tự 'A' đến 'F'
    j store_hex_char

convert_digit:
    addi $t3, $t3, '0'              # Chuyển số thành ký tự

store_hex_char:
    sb $t3, buffer($t2)             # Lưu ký tự vào buffer
    mflo $a0                        # Lấy phần nguyên của kết quả chia
    addi $t2, $t2, 1                # Tăng số lượng chữ số
    j reverse_hex_loop              # Lặp lại

reverse_hex_done:
    # Đảo chuỗi để đúng thứ tự
    li $t5, 0           # $t5 là chỉ số của chuỗi đảo ngược
reverse_hex_string_loop:
    blt $t5, $t2, continue_hex_reversing  # Nếu chưa xử lý hết chuỗi
    j reverse_hex_string_done              # Hoàn tất đảo chuỗi

continue_hex_reversing:
    sub $t6, $t2, $t5               # Tính vị trí đối xứng
```

```
    subi $t6, $t6, 1                # Điều chỉnh chỉ số
    lb $t7, buffer($t5)             # Lấy ký tự tại vị trí hiện tại
    lb $t8, buffer($t6)             # Lấy ký tự tại vị trí đối xứng
    sb $t7, buffer($t6)             # Hoán đổi các ký tự
    sb $t8, buffer($t5)
    addi $t5, $t5, 1                # Tăng chỉ số
    j reverse_hex_string_loop       # Lặp lại

reverse_hex_string_done:
    # Thêm ký tự kết thúc chuỗi
    sb $zero, buffer($t2)

    # Trở về chương trình chính
    jr $ra

# Hàm chuyển đổi số sang hệ 2
# Hàm chuyển đổi số sang hệ 2
convert_binary:
    li $t1, 16        # Đặt số lượng bit là 16
    li $t2, 0         # $t2 là chỉ số hiện tại trong buffer

convert_binary_loop:
    beq $t1, $zero, convert_binary_done  # Kiểm tra nếu đã xử lý hết 16 bit
    li $t3, 1
    sllv $t3, $t3, $t1               # Đặt bit 1 ở vị trí cần kiểm tra
    and $t3, $a0, $t3               # Kiểm tra bit tại vị trí đó
    beq $t3, $zero, store_zero      # Nếu bit là 0, lưu '0'
    addi $t3, $zero, '1'            # Nếu bit là 1, lưu '1'
    j store_binary_char

store_zero:
    addi $t3, $zero, '0'

store_binary_char:
    sb $t3, buffer($t2)             # Lưu ký tự vào buffer
    addi $t2, $t2, 1                # Tăng chỉ số buffer
    addi $t1, $t1, -1               # Giảm chỉ số bit
    j convert_binary_loop           # Lặp lại cho bit tiếp theo

convert_binary_done:
    # Thêm ký tự kết thúc chuỗi
    sb $zero, buffer($t2)

    # Trở về chương trình chính
    jr $ra
```

```
# Hàm ghi dữ liệu vào tập tin
# Hàm ghi dữ liệu vào tập tin với thông điệp mô tả
# $a1: địa chỉ của thông điệp mô tả
# $a2: địa chỉ của dữ liệu kết quả
write_file:
    # Ghi thông điệp mô tả vào tập tin
    li $v0, 15
    move $a0, $s6
    syscall

    # Ghi dữ liệu kết quả vào tập tin
    li $v0, 15
    la $a1, buffer
    syscall

    # Ghi dòng mới vào tập tin
    li $v0, 15
    la $a1, newline
    li $a2, 2
    syscall

    jr $ra
```