COMPUTER NETWORKS ASSIGNMENT 02

ESHITA KAPAT
23051671
CSE-28

5.) **PROGRAM:**

```
 2
 3 void check_endianness()
 4 {
 5 unsigned int num = 1;
 6 char *byte = (char*)&num;
 7
 8 if(*byte)
 9 printf("Host is Little endian");
10 else
11 printf("Host is Big endian");
12 }
13
14 void print_bytes(unsigned int num)
15 {
16 char *ptr = (char*)&num;
17 printf("Bytes:");
18 for(size_t i = 0 ; i < sizeof(num) ; i++)
19 printf("%02x" , ptr[i} & 0xFF);
20 printf("\n");
21 }
22
23 unsigned int reverse_endianness(unsigned int num)
24 {
25 unsigned int b0, b1 , b2, b3;
26 b0 = (num & 0x000000FF) << 24;
27     b1 = (num & 0x0000FF00) << 8;
28     b2 = (num & 0x00FF0000) >> 8;
29     b3 = (num & 0xFF000000) >> 24;
30     return (b0 | b1 | b2 | b3);
31     }
32
33     int main()
34     {
35 unsigned int num ,converted;
36
37 check_endianness();
```

C ▾ Tab Width: 8 ▾ Ln 41, Col 31 ▾ INS

```c
{
char *ptr = (char*)&num;
printf("Bytes:");
for(size_t i = 0 ; i < sizeof(num) ; i++)
printf("%02x" , ptr[i} & 0xFF);
printf("\n");
}

unsigned int reverse_endianness(unsigned int num)
{
unsigned int b0, b1 , b2, b3;
b0 = (num & 0x000000FF) << 24;
    b1 = (num & 0x0000FF00) << 8;
    b2 = (num & 0x00FF0000) >> 8;
    b3 = (num & 0xFF000000) >> 24;
    return (b0 | b1 | b2 | b3);
    }

    int main()
    {
unsigned int num ,converted;

check_endianness();
printf("Enter an unsigned integer :");
scanf("%u", &num);

printf("Original value: %u\n", num);
print_bytes(num);

converted = reverse_endianness(num);
printf("After endianness conversion: %u\n", converted);
print_bytes(converted);

return 0;
}
```

C ▾ Tab Width: 8 ▾ Ln 41, Col 31 ▾ INS

OUTPUT :

```
kiit@BT02793:~$ gcc program05.c -o program05
kiit@BT02793:~$ ./program05
Host is Little endianEnter an unsigned integer :5
Original value: 5
Bytes:05000000
After endianness conversion: 83886080
Bytes:00000005
```

4.) **PROGRAM**

Open | program04.c ~/ | Save

```c
#include<stdio.h>

struct pkt{
char ch1;
char ch2[2];
char ch3;
};

int main()
{
unsigned int num;
struct pkt packet;
unsigned int rebuilt;

printf("Enter a 4-byte integer (0 - 4294967295): ");
scanf("%u", &num);

    packet.ch1     = (num & 0xFF);
    packet.ch2[0]  = (num >> 8) & 0xFF;
    packet.ch2[1]  = (num >> 16) & 0xFF;
    packet.ch3     = (num >> 24) & 0xFF;

       printf("\nPacket contents:\n);
        printf("ch1     : 0x%02X\n", (unsigned char)packet.ch1);
    printf("ch2[0]  : 0x%02X\n", (unsigned char)packet.ch2[0]);
    printf("ch2[1]  : 0x%02X\n", (unsigned char)packet.ch2[1]);
    printf("ch3     : 0x%02X\n", (unsigned char)packet.ch3);

    rebuilt = ((unsigned char)packet.ch3   << 24)
             | ((unsigned char)packet.ch2[1]<< 16)
             | ((unsigned char)packet.ch2[0]<< 8)
             | ((unsigned char)packet.ch1);

            printf("\nReconstructed number: %u\n", rebuilt);

            return 0;
            }
```

OUTPUT :

```
kiit@BT02793:~$ gcc program04.c -o program04
kiit@BT02793:~$ ./program04
Enter a 4-byte integer (0 - 4294967295): 100

Packet contents:
ch1     : 0x64
ch2[0]  : 0x00
ch2[1]  : 0x00
ch3     : 0x00

Reconstructed number: 100
```

3.)

#include <stdio.h>

int main() {

```
    unsigned int num;
    char byte1, byte2, byte3, byte4;

    printf("Enter an unsigned integer: ");
    scanf("%u", &num);

    // Extract each byte using bitwise operations
    byte1 = (num & 0xFF);          // Least Significant Byte
    byte2 = (num >> 8) & 0xFF;
    byte3 = (num >> 16) & 0xFF;
    byte4 = (num >> 24) & 0xFF;    // Most Significant Byte

    printf("Byte 1 (LSB): 0x%02X\n", (unsigned char)byte1);
    printf("Byte 2      : 0x%02X\n", (unsigned char)byte2);
    printf("Byte 3      : 0x%02X\n", (unsigned char)byte3);
    printf("Byte 4 (MSB): 0x%02X\n", (unsigned char)byte4);

    return 0;
}
```

4.)

```
#include <stdio.h>
#include <string.h>

// Define the dob structure
struct dob {
    int day;
    int month;
    int year;
};

// Define the student_info structure
struct student_info {
    int roll_no;
    char name[50];
    float CGPA;
    struct dob age;
};

// Function that accepts student_info by value
void printStudentByValue(struct student_info s) {
    printf("Print by Value:\n");
    printf("Roll No: %d\n", s.roll_no);
    printf("Name   : %s\n", s.name);
    printf("CGPA   : %.2f\n", s.CGPA);
    printf("DOB    : %02d-%02d-%04d\n", s.age.day, s.age.month, s.age.year);
    printf("\n");
}
```

```
// Function that accepts student_info by address (pointer)
void printStudentByAddress(struct student_info *s) {
    printf("Print by Address:\n");
    printf("Roll No: %d\n", s->roll_no);
    printf("Name   : %s\n", s->name);
    printf("CGPA   : %.2f\n", s->CGPA);
    printf("DOB    : %02d-%02d-%04d\n", s->age.day, s->age.month, s->age.year);
    printf("\n");
}

int main() {
    struct student_info student;

    // Assign values
    student.roll_no = 101;
    strcpy(student.name, "John Doe");
    student.CGPA = 9.1;
    student.age.day = 15;
    student.age.month = 6;
    student.age.year = 2000;

    // Call functions
    printStudentByValue(student);
    printStudentByAddress(&student);

    return 0;
}

5.)  #include <stdio.h>
#include <stdlib.h>

// Function to swap two integers using pointers
void swap(int *a, int *b) {
    int temp;
    temp = *a;
    *a = *b;
    *b = temp;
}

int main(int argc, char *argv[]) {
    if (argc != 3) {
        printf("Usage: %s <num1> <num2>\n", argv[0]);
        return 1;
    }

    // Convert command line arguments to integers
    int x = atoi(argv[1]);
```

```
    int y = atoi(argv[2]);

    printf("Before swapping:\n");
    printf("x = %d, y = %d\n", x, y);

    // Swap using function with pointers
    swap(&x, &y);

    printf("After swapping:\n");
    printf("x = %d, y = %d\n", x, y);

    return 0;
}
```