ASSIGNMENT-1

1. Write a C program to swap the content of 2 variables entered through the command Line using function and pointer.

⇒

```
#Include <stdio.h>
void swap(int *p, int *q) {
        int temp = *p;
        *p = *q;
        *q = temp;
}
int main() {
        int a = 80;
        int b = 20;
        printf("Before swapping :\n");
        printf("After swap");
        printf("a= %d and b= %d \n", a, b);
        swap(&a, &b);
        printf("After swapping :\n");
        printf("a= %d and b= %d \n", a, b);
        return 0;
}
```

Output

Before swapping
a= 80, b= 20
After swapping
a=20, b= 80

2. Write a C program to assign values to each members of the following structure. Pass the populated structure to a function using call-by-value and another function using call-by-address and print the value of each member of the structure.

```
struct student_info {
        int roll_no;
        char name[50];
        float CGPA;
        struct dob age;
};
struct dob {
        int day;
        int month;
        int year;
};
```

⇒

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 2
struct dob
{
    int date;
    int month;
    int year;
};
struct stu_info
{
    int roll;
    char name[50];
    float cgpa;
    struct dob age;
};
```

```
void call_by_value (struct stu_info s)
{
    printf ("Enter Student details : \n");
    printf ("Enter Details of 2 Student : \n");
    printf ("ENTER ROLL : ");
    scanf ("%d", &s.roll);
    printf ("ENTER NAME : ");
    scanf ("%s", s.name);
    printf ("ENTER CGPA : ");
    scanf ("%f", &s.cgpa);
    printf ("Enter date month year of your
birthday : ");
    scanf ("%d", &s.age.date);
    scanf ("%d", &s.age.month);
    scanf ("%d", &s.age.year);
}
void call_by_address (struct stu_info *s)
{
    printf ("Enter Student Details : \n");
    printf ("Enter details of 2 student : \n");
    printf ("Enter Roll : ");
    scanf ("%d", &s->roll);
    printf ("Enter name : ");
    scanf ("%s", s->name);
    printf ("Enter CGPA:");
    scanf ("%f", &s->cgpa);
    printf ("Enter date month year of your
birthday : ");
```

```
    scanf("%d", &s->age.date);
    scanf("%d", &s->age.month);
    scanf("%d", &s->age.year);
}
void print_details(struct stu_info s)
{
    printf("Student Details:\n");
    printf("Details of Student:\n");
    printf("ROLL: %d\n", s.roll);
    printf("NAME: %s\n", s.name);
    printf("CGPA: %.2f\n", s.cgpa);
    printf("BIRTH DATE: %d/%d/%d\n",
                                s.age.date,
                    s.age.month, s.age.year);
    print("\n");
}
int main()
{
    struct stu_info s1, s2;
    call_by_value(s1);
    call_by_address(&s2);
    print_details(s1);
    print_details(s2);
    return 0;
}
```

Output

```
Enter Student Details :
Enter Details of 2 Student :
ENTER ROLL : 583
ENTER NAME : Souvik
ENTER CGPA : 8.5
ENTER DATE MONTH YEAR OF YOUR BIRTHDAY : 10.05.2003
ENTER STUDENT DETAILS :
ENTER ROLL : 154
ENTER NAME : RAVI
ENTER CGPA : 8.7
ENTER DATE MONTH YEAR OF YOUR BIRTHDAY : 11:07.2003
STUDENT DETAILS :
ROLL : 583
NAME : Souvik
CGPA : 8.5
BIRTHDATE : 10/05/2003
STUDENT DETAILS :
ROLL : 154
NAME : RAVI
CGPA : 8.7
BIRTHDATE : 11/07/2003
```

3. Write a C program to extract each byte from a given number and store them in seperate character variables and print the content of through variables.

⇒

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int n = 258;
    unsigned char *ptr = (unsigned char*) &n;
    int *p = &n;
    int firstbyte = (n & 255);
    int secondbyte = (n & 65280);
    secondbyte = secondbyte >> 8;
    int thirdbyte = (n & 0x00ff0000);
    thirdbyte = thirdbyte >> 16;
    int fourthbyte = (n & 0xff000000);
    fourthbyte = fourthbyte >> 24;
    printf ("\n\nvalue of the bytes : %d %d %d %d
%d\n", fourthbyte, thirdbyte, secondbyte, firstbyte);
    return 0;
}
```

Output

value of the bytes : 0 0 1 2

4. Write a C program to enter a number and store the number across the following structure and print the content of each member of the structure. Then aggregate each member of the structure to form the original number and print the same.

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

Source code:

```
#include <stdio.h>
struct pkt
{
    char ch1;
    char ch[2];
    char ch3;
};
int main()
{
    int num;
    scanf("%d",&num);
    int n = num;
    int b1, b2, b3, b4;
    b1 = num%256;
    num = num/256;
    b2 = num%256;
    num = num/256;
    b3 = num%256;
    num = num/256;
    b4 = num;
```

```
printf("digit in first byte = %d\n", b1);
printf("digit in second byte = %d\n", b2);
printf("digit in third byte = %d\n", b3);
printf("digit in fourth byte = %d\n", b4);
struct pkt p;
p.ch1 = b1;
p.ch[0] = b2;
p.ch2[0] = b3;
p.ch3 = b4;
printf("1st member of the structure = %d\n", p.ch1);
printf("2nd member of the structure = %d %d\n",
                    p.ch2[0], p.ch2[1]);
printf("3rd member of the structure = %d\n", p.ch3);
printf("Regenerated number = %d", n);
}
```

Output

258

digit in first byte = 2
digit in second byte = 1
digit in third byte = 0
digit in fourth byte = 0
1st member of the structure = 2
2nd members of the structure = 1, 0
3rd member of the structure = 0
the regenerated number = 258

5. Write a C program to check whether the Host Machine is in Little Endian or Big Endian. Enter a number, print the constant of each byte location and convert the endianess of the same i.e Little to big endian and vice versa.

⇒

```
#include <stdio.h>
int main()
{
    int num;
    scanf("%d", &num);
    int i = 1, b1, b2, b3, b4;
    int n = num;
    b1 = num % 256;
    num = num / 256;
    b2 = num % 256;
    num = num / 256;
    printf("Extracted byte from the LSB of the no. = %d\n", b1);
    printf("Extracted byte from the LSB of the no. = %d\n", b2);
    printf("Memory represented of the no.\n");
    printf("-------------\n");
    printf("Memory Address → value\n");
    printf("%d \t → \t %d\n", &b1, b1);
    printf("%d \t → \t %d\n", &b2, b2);
    printf("%d \t → \t %d\n", &b3, b3);
    printf("%d \t → \t %d\n", &b4, b4);
    if (b1 == *(char*)&n)
    {
        printf("The LSB is stored at the lowest memory address\n, Hence the Host machine is Little Endian\n");
        printf("The number is converted to Big Endian\n
                Memory represented of the Number\n");
```

```
else
{
printf("The LSB is stored at the highest memory address.\n. Hence the host machine is in Big Endian");
printf(" the no. is converted to Little Endian mode\n
Memory representation of No.\n");
printf("- - - - - - - - - - \n");
printf("Memory Address → Value\n");
printf("- - - - - - - - - - \n");
char cbyte[4];
cbyte[0] = b4;
cbyte[1] = b3;
cbyte[2] = b2;
cbyte[3] = b1;
printf("%d\t→\t%d\n", &cbyte[0], b4);
printf("%d\t→\t%d\n", &cbyte[1], b3);
printf("%d\t→\t%d\n", &cbyte[2], b2);
printf("%d\t→\t%d\n", &cbyte[3], b1);
return 0;
}
```

Output

```
258
extracted byte from the LSB of the no. = 2
extracted byte from the LSB of the no. = 1
Memory represented of no.
- - - - - - - - - - - - - - - - - - - -
Memory Address → Value
- - - - - - - - - - - - - - - - - - - -
6422292 → 2
6422288 → 1
6422284 → 0
6422280 → 0
```

The LSB is stored at the lowest memory addrn
Hence, the host machine is in Little Endian
The number is converted to Big Endian now.

Memory representation of the Number

Memory Address	→	Value
6422272	→	0
6422273	→	0
6422274	→	1
6422275	→	2

7/8/23

ASSIGNMENT-2

1. Write a sender and receiver program in C by passing the IP address and the port number of each other.

⇒ receiver.c :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <string.h>
#include <unistd.h>
int main()
{
    int sock = socket(AF_INET, SOCK_DGRAM, 0);
    if (sock == -1)
    {
        printf("file couldn't be created");
        exit(1);
    }
    struct sockaddr_in recv_soc;
    recv_soc.sin_family = AF_INET;
    recv_soc.sin_port = 5000;
    recv_soc.sin_addr.s_addr = INADDR_ANY;
    int r = bind(sock, (const struct sockaddr *)
&recv_soc, sizeof(recv_soc));
    if (r == -1)
    {
        printf("Error in bind...");
        exit(1);
    }
```

```
char recv_buff[100];
struct sockaddr_in srcaddr;
int len = sizeof(srcaddr);
int recv = recvfrom(sock, recv_buff, sizeof(recv_buff), 0, (struct sockaddr*)&srcaddr, &len);
if(recv==-1)
{
    printf("Error when receiving");
    exit(1);
}
for(int i=0; i<len; i++)
{
    printf("%c", recv_buff[i]);
}
//send to API
struct sockaddr_in recv_soc2;
recv_soc2.sin_family = AF_INET;
recv_soc2.sin_port = 4000;
recv_soc2.sin_addr.s_addr = INADDR_ANY;
char str[100];
strcpy(str, "Hello");
int l = strlen(str);
sendto(sock, str, l, 0, (const struct sockaddr*)
&recv_soc2, sizeof(recv_soc2));
// close API
int c = close(sock);
if(c==-1)
{
    printf("Error in closing file");
    exit(1);
}
return 0;
}
```

sender.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <string.h>
#include <unistd.h>
int main()
{
  int sock= socket (Af_INET, SOCK_DGRAM, 0);
  if (sock == -1)
  {
     printf ("file couldnt be created.");
     exit (1);
  }
  struct sockaddr_in recv_soc;
  recv_soc.sin_family = Af_INET;
  recv_soc.sin_port)=4000;
  recv_soc.sin_addr.s_addr = INADDR_ANY;
  int r= bind (sock, (const struct sockaddr*) &recv_soc,
sizeof (recv_soc));
  if (r== -1)
  {
     printf ("Error in bind");
     exit (1);
  }
  struct sockaddr_in recv_soc1;
  recv_soc1.sin_family = Af_INET;
  recv_soc1.sin_port)=5000;
  recv_soc1.sin_addr.s_addr = INADDR_ANY;
  char str [100];
  strcpy (str, "Hello");
  int l = strlen (str);
  sendto (sock, str, l, 0, (const struct sockaddr*)& recv_soc1,
```

```
sizeof(recv_so1)),
char recv_buff[100];
struct sockaddr srcaddr;
int len = sizeof(srcaddr);
int recv = recvfrom(sock, recv_buff, sizeof(recv_buff),
0, (struct sockaddr *)& srcaddr, & len);
if(recv == -1)
{
    printf("Error when receiving");
    exit(1);
}
int c = close(sock);
if(c == -1)
{
    printf("Error in closing file");
    exit(1);
}
return 0;
```

21/8/23

OUTPUT:

RECEIVER

```
gcc receiver.c -o receive
./receive
Hello there....
```

SENDER

```
gcc sender.c -o send
./send
Hello there....
```

Name- Sourik Basak
Roll no - 21055 83

Assignment - 3

1. Write a program (client & server) to interact with each other.

Source code:

client.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <string.h>
#include <unistd.h>
int main()
{
    int sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        printf("Error in creation of socket");
        exit(1);
    }
    struct sockaddr_in server_sock;
    server_sock.sin_family = AF_INET;
    server_sock.sin_port = 6300;
    server_sock.sin_addr.s_addr = INADDR_ANY;
    int len = sizeof(server_sock);
```

```
int con = connect (sockfd, (const struct sockaddr*)
  & server-sock, len);
char str[100];
printf ("Enter a message for server: ");
scanf ("%s", str);
int send2 = send (sockfd, str, sizeof (str), 0);
if (send2 == -1) {
    printf ("Error in send");
    exit (1);
}
int recv2 = recv (sockfd, str, sizeof (str), 0);
if (recv2 == -1) {
    printf ("Error in recv");
    exit (1);
}
printf ("Message from server: %s", str);
close (sockfd);
```

server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <string.h>
#include <unistd.h>
int main() {
    int sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1) {
        print(" Error in creation of socket ");
        exit(1);
    }
    struct sockaddr_in server_sock;
    server_sock.sin_family = AF_INET;
    server_sock.sin_port = 6300;
    server_sock.sin_addr.s_addr = INADDR_ANY;
    int rct = bind(sockfd, (const struct sockaddr*)
& server_sock, (sizeof(server_sock)));
    if(rct == -1){
        printf("Error in bind");
    }
    int lis = listen(sockfd, 5);
    struct sockaddr_in client_sock;
    int len = sizeof(client_sock);
    int datafd = accept(sockfd, (struct sockaddr *) &
client_sock, &len);
    if (datafd == -1) {
        printf("Error in c-sock");
        exit(1); }
```

```
char str[100];
int recv1 = recv(datafd, str, sizeof(str), 0);
if (recv1 == -1) {
    printf("Error in recv");
    exit(1);
}
printf("Message from client : %s", str);
printf("Enter message for client:");
scanf("%s", str);
int send1 = send(datafd, str, sizeof(str), 0);
if (send1 == -1) {
    printf("Error in send");
    exit(1);
}
close(sockfd);
close(datafd);
```

Output

client.c

```
./client
Enter message for server :
hello
Message from server :
hii
```

server.c

```
./server
Message from client :
hello
Enter message for client :
hii
```

Name- Souvik Basak

Rollno- 2105583

ASSIGNMENT - 4

Create server and client programs to communicate via cables using IP Address.

Source code:

client.c

```
#include<stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
int main(){
    int sockfd = socket(AF_INET,SOCK,SOCK_STREAM,0);
    if(sockfd==-1){
        printf("Error in creation of socked");
        exit(1)
    }
    struct sockaddr_in c_sock;
    c_sock.sin_family = AF_INET;
    c_sock.sin_port = htons(8200);
    int inettt = inet_atom("192.168.183.92", &c_sock.
                    sin_addr);
    int len= size_of(c_sock);
    int conn = connect(sockfd,(const struct sockaddr*)
        &c_sock, len);
    char str[100];
```

server.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>
int main()
{
    struct sockaddr_in server_sock;
    server_sock.sin_family = AF_INET;
    server_sock.sin_port = htons(8200);
    inet_aton("192.168.183.92", &server_sock.sin_addr);
    int server_sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if(server_sockfd == -1)
    {
        printf("Error in creation of socket");
        exit(1);
    }
    int ret = bind(server_sockfd, (const struct sockaddr *)
&server_sock, sizeof(server_sock));
    if(ret == -1)
    {
        printf("Error in bind");
        exit(1);
    }
    int lis = listen(server_sockfd, 5);
    char str[100];
    while(1)
    {
        struct sockaddr_in client_sock;
        int size = sizeof(client_sock);
```

```
int datafd1 = accept (server_sockfd, (struct
sockaddr *) &client_sock, &size);
if (datafd == -1)
{
    printf ("\n Error in making dedicated
    connection.");
    exit(1);
}
int recv = recv (datafd1, str, sizeof (str), 0);
if (rev == -1)
printf ("\n Error in recv");
printf ("Message from client : %s\n", str);
close (datafd1);
}
close (server_sockfd);
return 0;
```

Output

server

```
gcc server.c -o send
./send.out
Hey
Enter message : Hello
Hi
Enter message : Bye
```

client 1

```
gcc client.c -o client1
./client1.out
Enter message for
server : Hey
Hello
```

client 2

```
gcc client2.c client2
./client2.out
Enter message for
server : Hi
Bye
```

Assignment - 5

1. Write a code to create a sequential server serving multiple clients at a time.

server.c

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main()
{
    int sockfd = socket (AF_INET, SOCK_STREAM, 0);
    if (sockfd == -1)
    {
        printf (" Error in socket creation \n ");
        return 0;
    }
    struct sockaddr_in recv_sock;
    recv_sock.sin_family = AF_INET;
    recv_sock.sin_port = 4150;
    recv_sock.sin_addr.s_addr = INADDR_ANY;
    int ret_bind = bind (sockfd, (const struct sockaddr*)
     & recv_sock, sizeof (recv_sock));
```

```
if(ret_bind == -1)
{   printf("Error in binding the socket\n");
    return 0;
}
int ret_lisb = listen(sockfd, 5)
if(ret_lis == -1)
{   printf("Error in bind listen");
    return 0;
}
while(1) {
    int len_recv_sock = sizeof(recv_sock);
    int datafd = accept(sockfd, (struct sockaddr)* &
recv_sock, &len_recv_sock);
    if(datafd == -1)
    {   printf("Error in accept\n");
        return 0;
    }
    int ret = fork();
    if(ret == 0) {
        close(sockfd);
    char recv_buff[100];
    struct sockaddr_in src_addr;
    int len = sizeof(src_addr);
    int ret_recv = recv(datafd, recv_buff,
sizeof(recv_buff), 0);
    if(ret_recv == -1)
    printf("Error in receiving");
```

```
char send_str[100];
printf("<----.");
scanf("%s", send_str);
int ret_send = send(datafd, send_str, strlen(
        send_str), 0);
if (ret_send == -1)
{
    printf("Error in sending to\n");
    return 0;
}
break;
}
if (ret != 0)
close(datafd);
}
return 0;
}
```

client.c

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

int main()
{
    int sockfd = socket(AF_INET, SOCK_STREAM, 0)
    if (sockfd == -1)
    printf("Error in socket");
    struct sockaddr_in recv_sock;
    recv_sock.sin_family = AF_INET;
    recv_sock.sin_port = 4150;
    recv_sock.sin_addr.s_addr = inaddr_ANY;
    if ret_con = connect(sockfd, (const struct sockaddr*)
& recv_sock, sizeof(recv_sock));
    if (ret_con == -1)
    {
        printf("Error in connection");
        return 0;
    }
    char recv_buff[100];
    struct sockaddr_in src_addr;
    int len = sizeof(src_addr);
    int k=3;
    while(k>=0){
        k--;
```

```
char send_str[100];
printf("<---");
scan("%s", send_str);
int ret_send= send(sockfd, send_str, strlen(str,0);
if (ret_send ==-1)
printf("Error in sending");
char recv_buff[100];
struct sock addr_in src_addr;
int len = sizeof(src_addr)
int ret_recv = recv(sockfd, recv buff, sizeof(recv_
                                              buff),0)
if (ret_recv ==-1)
print("Error in ~~returning~~"); receiving");
printf("---> %s\n", recv_buff);
if (recv_buff == "bye")
break;
}
close(sock fd);
return 0;
```

Output

Server	Client 1	Client 2
gcc servers.c -o server ./server Msg from client Hi from client 2 Msg for client Hi from client 1	gcc client1.c -o client1 ./client 1 Hello Client	gcc client2.c- o client 2 ./client 2 Hello Client

Name- Souvik Basak
Rollno- 2105583

Assignment - 6

creating client and server programs to receive and send the list of the files in the current directory respectively.

client.c

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
int main(){
  int sock = socket(AF_INET, SOCK_STREAM, 0);
  struct sockaddr_in addr;
  addr.sin_family = AF_INET;
  addr.sin_port = 8040;
  addr.sin_addr.s_addr = INADDR_ANY;
  int cn = connect(sock, (const struct sockaddr*)&addr,
                   sizeof(addr));
  FILE *f = fopen("res.txt", "w");
  while(1)
  {
    char recv_buff[100];
    int rv = recv(sock, recv_buff, sizeof(recv_buff), 0);
    int size = fwrite(recv_buff, 5, 100, f);
    if(size<100)
      break;
  } return 0;
```

server.c

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <unistd.h>
main()
int sock = socket(AF_INET, SOCK_STREAM, 0)
struct sockaddr_in recv_soc;
recv_soc.sin_family = AF_INET;
recv_soc.sin_family = 5000;
recv_soc.sin_addr.s_addr = INADDR_ANY;
int = bind(sock, (const struct sockaddr*) &recv_soc,
      sizeof(recv_soc));
int l = listen(sock, 5);
struct sockaddr_in sock_data;
int len = sizeof(sock_data);
int datafd = accept(sock, (struct sockaddr*) &sock_data,
      &len);
int sys = system("ls>re.txt");
FILE* f = fopen("res.txt", "r");
while(feof(f))
{   char str[100];
    int size = fread(str, 5, 100, f);
    int l = strlen(str);
    int s = send(sock, str, l, 0);
}
close(sock);
return 0;
```

OUTPUT

server output

listening....
file received and saved as
"res.txt"

client output

file data sent successfully

6/11/23

Sourik Basak
210583