TASK_1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/ipc.h>
#include <sys/shm.h>

struct shared {
   char sel[100];
   int b;
};

int main() {
   int pipefd[2];
   pid_t pid;
   key_t key = 1234;
   int shmid;
   struct shared *shm;

   shmid = shmget(key, sizeof(struct shared), 0666 | IPC_CREAT);        //shared mem
   if (shmid == -1){
      perror("shmget failed");
      exit(1);
   }

   shm = (struct shared *)shmat(shmid, NULL, 0);
   if (shm == (void *) - 1) {
      perror("shmat failed");
      exit(1);
   }

   shm->b = 1000;

   if (pipe(pipefd) == -1){
      perror("pipe failed");
      exit(1);
   }

   pid = fork();

   if (pid < 0){
      perror("fork failed");
      exit(1);
   }
```

```
if (pid > 0){
   close(pipefd[0]);

   printf("Provide your input from given options:\n");
   printf("1. Type a to add money\n");
   printf("2. Type w to withdraw money\n");
   printf("3. Type c to check balance\n");

   scanf("%s", shm->sel);
   printf("Your selection: %s\n", shm->sel);


   wait(NULL);

   char buffer[100];
   read(pipefd[0], buffer, sizeof(buffer));
   printf("%s\n", buffer);
   shmdt(shm);
   shmctl(shmid, IPC_RMID, NULL);
   close(pipefd[1]);
}

else{
   close(pipefd[1]);
   if (strcmp(shm->sel, "a") == 0){
      int amount;
      printf("\nEnter amount to be added:\n");
      scanf("%d", &amount);

      if (amount > 0) {
         shm->b += amount;
         printf("Balance added successfully\n");
         printf("Updated balance after addition:\n%d\n", shm->b);
      } else {
         printf("Adding failed, Invalid amount\n");
      }

   }else if (strcmp(shm->sel, "w") == 0){
      int amount;
      printf("\nEnter amount to be withdrawn:\n");
      scanf("%d", &amount);

      if (amount > 0 && amount <= shm->b){
         shm->b -= amount;
         printf("Balance withdrawn successfully\n");
         printf("Updated balance after withdrawal:\n%d\n", shm->b);
```

```
        }
        else{
            printf("Withdrawal failed, Invalid amount\n");
        }

    }
    else if (strcmp(shm->sel, "c") == 0){
        printf("\nYour current balance is:\n%d\n", shm->b);

    }
    else{
        printf("\nInvalid selection\n");
    }

    write(pipefd[1], "Thank you for using", 19);
    close(pipefd[0]);
    exit(0);
  }

  return 0;
}
```

Output

```
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ gcc_task1.c -o task1
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task1
Provide your input from given options:
1. Type a to add money
2. Type w to withdraw money
3. Type c to check balance
Invalid selection
a
Your selection: a
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task1
Provide your input from given options:
1. Type a to add money
2. Type w to withdraw money
3. Type c to check balance
Invalid selection
W
Your selection: w
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task1
Provide your input from given options:
1. Type a to add money
2. Type w to withdraw money
3. Type c to check balance
Invalid selection
```

```
c
Your selection: c
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task1
Provide your input from given options:
1. Type a to add money
2. Type w to withdraw money
3. Type c to check balance
w
Your selection: w
Enter amount to be withdrawn:
300
Balance withdraw successfully
Updated balance after withdrawal:
700
Thank you for using
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$
```

TASK_2

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <sys/wait.h>

struct msg {
   long int type;
   char txt[6];
};

void login_process(int msgid) {
   struct msg message;
```

```
    char workspace[10];

    printf("Please enter the workspace name:\n");
    scanf("%s", workspace);

    if (strcmp(workspace, "cse321") != 0) {
        printf("Invalid workspace name\n");
        exit(0);
    }

    message.type = 1;
    strcpy(message.txt, workspace);
    msgsnd(msgid, &message, sizeof(message.txt), 0);
    printf("Workspace name sent to OTP generator from log in: %s\n", message.txt);
}

void otp_generator_process(int msgid) {
    struct msg message;

    msgrcv(msgid, &message, sizeof(message.txt), 1, 0);
    printf("OTP generator received workspace name from log in: %s\n", message.txt);

    pid_t otp = getpid();
    snprintf(message.txt, sizeof(message.txt), "%d", otp);

    message.type = 2;
    msgsnd(msgid, &message, sizeof(message.txt), 0);
    printf("OTP sent to log in from OTP generator: %s\n", message.txt);

    message.type = 3;
```

```
    msgsnd(msgid, &message, sizeof(message.txt), 0);
    printf("OTP sent to mail from OTP generator: %s\n", message.txt);
}

void mail_process(int msgid) {
    struct msg message;

    msgrcv(msgid, &message, sizeof(message.txt), 3, 0);
    printf("Mail received OTP from OTP generator: %s\n", message.txt);

    message.type = 4;
    msgsnd(msgid, &message, sizeof(message.txt), 0);
    printf("OTP sent to log in from mail: %s\n", message.txt);
}

int main() {
    key_t key;
    int msgid;
    pid_t otp_gen_pid, mail_pid;
    struct msg message1, message2;

    key = ftok("progfile", 65);
    msgid = msgget(key, 0666 | IPC_CREAT);

    otp_gen_pid = fork();
    if (otp_gen_pid == 0) {
        otp_generator_process(msgid);
        exit(0);
    } else {
        mail_pid = fork();
```

```
        if (mail_pid == 0) {
            mail_process(msgid);
            exit(0);
        } else {
            login_process(msgid);

            wait(NULL);
            wait(NULL);

            msgrcv(msgid, &message1, sizeof(message1.txt), 2, 0);
            printf("Log in received OTP from OTP generator: %s\n", message1.txt);

            msgrcv(msgid, &message2, sizeof(message2.txt), 4, 0);
            printf("Log in received OTP from mail: %s\n", message2.txt);

            if (strcmp(message1.txt, message2.txt) == 0) {
                printf("OTP Verified\n");
            } else {
                printf("OTP Incorrect\n");
            }
            msgctl(msgid, IPC_RMID, NULL);
        }
    }

    return 0;
}
```

Output

```
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ gcc task2.c -o task2
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task2
Please enter the workspace name:
CSE321
Invalid workspace name
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task2
Please enter the workspace name:
abc
Invalid workspace name
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$ ./task2
Please enter the workspace name:
cse321
Workspace name sent to OTP generator from log in: cse321
OTP generator received workspace name from log in: cse321
OTP sent to log in from OTP generator: 5366
OTP sent to mail from OTP generator: 5366
Mail received OTP from OTP generator: 5366
OTP sent to log in from mail: 5366
lubna@Lubna-HP-Pavilion-Laptop-15-cs2xxx:~$
```