TASK-5A

```
 #include<stdio.h>
#include<unistd.h>
int main() {
int pipefds[2];
int returnstatus;
char writemessages[2][20]={“Hi”, “Hello”};
char readmessage[20];
returnstatus = pipe(pipefds);
if (returnstatus == -1) {
printf(“Unable to create pipe\n”);
return 1;
}
printf(“Writing to pipe – Message 1 is %s\n”, writemessages[0]);
write(pipefds[1], writemessages[0], sizeof(writemessages[0]));
read(pipefds[0], readmessage, sizeof(readmessage));

printf(“Reading from pipe – Message 1 is %s\n”, readmessage);
printf(“Writing to pipe – Message 2 is %s\n”, writemessages[0]);
write(pipefds[1], writemessages[1], sizeof(writemessages[0]));
read(pipefds[0], readmessage, sizeof(readmessage));
printf(“Reading from pipe – Message 2 is %s\n”, readmessage);
return 0;
}
```

TASK-5B

```
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;
shmid=shmget((key_t)2345, 1024, 0666|IPC_CREAT);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);
}
```