1.c

```
#include<stdio.h>
#include<pthread.h>
#include<unistd.h>
#include<stdlib.h>

struct arg_struct {

int a1;
int a2;
int a3;
int a4;
int a5;
};

void *arguments(void *args)
{
struct arg_struct *a=args;
float c=((a->a1)+(a->a2)+(a->a3)+(a->a4)+(a->a5))/5;
printf("average marks of student is : ");
printf("%f",c);
}

int main()

{
pthread_t t;
int p,q,r,s,t1;

printf("enter marks in math ");
scanf("%d",&p);
printf("enter marks in science ");
scanf("%d",&q);
printf("enter marks in eng ");
scanf("%d",&r);
printf("enter marks in computer ");
scanf("%d",&s);
printf("enter marks in social science ");
scanf("%d",&t1);

struct arg_struct args={p,q,r,s,t1};
pthread_create(&t,NULL,&arguments,&args);
pthread_join(t,NULL);

}
```

2.c

```
#include<stdio.h>
#include<pthread.h>
#include<unistd.h>
#include<stdlib.h>

struct arg_struct1 {
int a1;
int a2;
};
struct arg_struct2 {
int a3;
int a4;
};

void *arguments1(void *args)
{
struct arg_struct1 *b=args;
int c=((b->a1)+(b->a2));
printf("Addition of two no.  is : ");
printf("%d",c);
}

void *arguments2(void *args1)
{
struct arg_struct2 *a=args1;
int d=((a->a3)*(a->a4));
printf(" Multiplication  of two no.  is :");
printf("%d",d);
}

int main()

{
pthread_t T1,T2;
int p,q,r,s;

printf("enter the two numbers to be added :\n");
scanf("%d%d",&p,&q);
printf("enter the two numbers to be multiplied :\n");
scanf("%d%d",&r,&s);

struct arg_struct1 args={p,q};
struct arg_struct2 args1={r,s};
pthread_create(&T1,NULL,&arguments1,&args);
pthread_join(T1,NULL);
pthread_create(&T2,NULL,&arguments2,&args1);
pthread_join(T2,NULL);


}
```

3.c

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<pthread.h>
#include<semaphore.h>
void *fun1();
void *fun2();
int sh=3;
sem_t a;
int main()
{
printf("initial value of sh is %d ",sh);
sem_init(&a,0,1);
pthread_t thread1,thread2;
pthread_create(&thread1,NULL,fun1,NULL);
pthread_create(&thread2,NULL,fun2,NULL);
pthread_join(thread1,NULL);
pthread_join(thread2,NULL);
printf("final value of sh is %d ",sh);
}
void *fun1()
{
int x;
sem_wait(&a);
x=sh;
x++;
sleep(1);
sh=x;
sem_post(&a);
}

void *fun2()
{
int y;
sem_wait(&a);
y=sh;
y--;
sleep(1);
sh=y;
sem_post(&a);
}
```

4.c

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>

typedef struct {

  int position;
  int count;
  sem_t *forks;
  sem_t *lock;
} test_t;

void initialize_semaphores(sem_t *lock, sem_t *forks, int num_forks);

void run_all_threads(pthread_t *threads, sem_t *forks, sem_t *lock, int num_philosophers);

void *philosopher(void *test);

void think(int position);

void eat(int position);

int main(int argc, char *args[])

{

  int num_philosophers = 5;

  sem_t lock;

  sem_t forks[num_philosophers];

  pthread_t philosophers[num_philosophers];

  initialize_semaphores(&lock, forks, num_philosophers);

  run_all_threads(philosophers, forks, &lock, num_philosophers);

  pthread_exit(NULL);

}
void initialize_semaphores(sem_t *lock, sem_t *forks, int num_forks)

{
  int i;

  for(i = 0; i < num_forks; i++) {

    sem_init(&forks[i], 0, 1);
```

```
  }
  sem_init(lock, 0, num_forks - 1);

}

void run_all_threads(pthread_t *threads, sem_t *forks, sem_t *lock, int num_philosophers)

{
  int i;

  for(i = 0; i < num_philosophers; i++) {

    test_t *arg = malloc(sizeof(test_t));
    arg->position = i;

    arg->count = num_philosophers;

    arg->lock = lock;

    arg->forks = forks;

    pthread_create(&threads[i], NULL, philosopher, (void *)arg);

  }

}

void *philosopher(void *test)

{
  int i;
   test_t self = *(test_t *)test;

  for(i = 0; i < 3; i++) {
 for(i = 0; i < 3; i++) {

    think(self.position);

    sem_wait(self.lock);

    sem_wait(&self.forks[self.position]);

    sem_wait(&self.forks[(self.position + 1) % self.count]);

    eat(self.position);

    sem_post(&self.forks[self.position]);

    sem_post(&self.forks[(self.position + 1) % self.count]);

    sem_post(self.lock);
```

```
  }
  think(self.position);
  pthread_exit(NULL);
}
void think(int position)
{
  printf("Philosopher %d thinking...\n", position);
}
void eat(int position)
{
  printf("Philosopher %d eating...\n", position);
}
```