Q5	Write C program that solves the Readers-Writers-Problem.
Code	

emad@vbox: ~/Desktop/emad1007395/assignment5/Q5

GNU nano 2.5.3 File: q5.c

```
#include<stdio.h>
#include<conio.h>
#include<stdbool.h>

struct semaphore {
int mutex;
int rcount;
int rwait;
bool wrt; };

voidaddR(struct semaphore *s) {
if (s->mutex == 0 && s->rcount == 0) {
printf("\nSorry, File open in Write mode.\nNew Reader added to $
s->rwait++;
} else {
printf("\nReader Process added.\n");
s->rcount++;
s->mutex--;
}
return;
}

voidaddW(struct semaphore *s) {
if(s->mutex==1) {
s->mutex--;
s->wrt=1;
printf("\nWriter Process added.\n");
}

else if(s->wrt) printf("\nSorry, Writer already operational.\n"$
elseprintf("\nSorry, File open in Read mode.\n");
return ;
}


^G Get Help ^O Write Out^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File^\ Replace ^U Uncut Tex^T To Spell
```

```
}

voidremR(struct semaphore *s) {
if(s->rcount == 0) printf("\nNo readers to remove.\n");
else {
printf("\nReader Removed.\n");
s->rcount--;
s->mutex++;
}
return ;
}

voidremW(struct semaphore *s) {
if(s->wrt==0) printf("\nNo Writer to Remove");
else {
printf("\nWriter Removed\n");
s->mutex++;
s->wrt=0;
if(s->rwait!=0) {
s->mutex-=s->rwait;
s->rcount=s->rwait;
s->rwait=0;
printf("%d waiting Readers Added.",s->rcount);
}
}
}

int main() {
struct semaphore S1={1,0,0};
while(1) {
system("cls");
printf("Options :-\n1.Add Reader.\n2.Add Writer.\n3.Remove Read$
intch;
scanf("%d",&ch);
```

```
emad@vbox: ~/Desktop/emad1007395/assignment5/Q5
  GNU nano 2.5.3            File: q5.c

intch;
scanf("%d",&ch);
switch(ch) {
case 1: addR(&S1); break;
case 2: addW(&S1); break;
case 3: remR(&S1); break;
case 4: remW(&S1); break;
case 5: printf("\n\tGoodBye!"); getch(); return 0;
default: printf("\nInvalid Entry!"); continue;
}

printf("\n\n<<<<<< Current Status >>>>>>\n\n\tMutex\t\t:\t%d\n\$
system("pause");
}
}
```