```
code Main

  -- OS Class: Project 3
  --
  -- AMANDEEP KAUR
  --

-----------------------------  Main  ---------------------------------

  function main ()

    InitializeScheduler()
    --SleepingBarberTester()
    SleepingBarberTester2()

    ThreadFinish()

    endFunction

-----------------------Sleeping Barber Problem------------------------
const
   CHAIRS = 5
   NumOfCustomers=10

var
    customer:Semaphore= new Semaphore
    barber:Semaphore= new Semaphore
    lock:Mutex=new Mutex
    numwait:int=0  -- number of customers in shop (including waiting and taking cut)
    hairCutFinished:bool = false
    printLock:Mutex=new Mutex
    --bthread:Thread=new Thread           --barber thread
    --status:char              --status of customer

function SleepingBarberTester ()
    var
--       NumOfCustomers:int=10
       i:int
       tharray: array [NumOfCustomers] of Thread = new array of Thread{NumOfCustomers
of new Thread}                    --array of 10 customers
       bthread:Thread

    bthread =  new Thread

    print("       Barber  1  2  3  4  5  6  7  8  9  10")
    printChar ('\n')
    lock.Init()                   --initailize mutex lock
    printLock.Init()
                   --initialize barber thread
    customer.Init(0)
barber.Init(0)

    bthread.Init("Barber")
    bthread.Fork(Barber,1)

    for (i=0; i<7; i=i+1)
        tharray[i].Init("Customer")
        tharray[i].Fork(Customer,i)
        currentThread.Yield()
    endFor

    for (i=0; i<1000; i=i+1)
    endFor


    for (i=7; i < 10; i=i+1)
        tharray[i].Init("Customer")
        tharray[i].Fork(Customer, i)
        currentThread.Yield()
    endFor

    ThreadFinish()
```

```
endFunction

function SleepingBarberTester2()
     var
--       NumOfCustomers:int=10
       i:int
       tharray: array [NumOfCustomers] of Thread = new array of Thread{NumOfCustomers
of new Thread}                      --array of 10 customers
       bthread:Thread

    bthread =  new Thread

    print("       Barber  1  2  3  4  5  6  7  8  9  10")
    printChar ('\n')
    lock.Init()                  --initaizlize mutex lock
    printLock.Init()
                   --initialize barber thread
    customer.Init(0)
    barber.Init(0)

    bthread.Init("Barber")
    bthread.Fork(Barber,1)

    for (i=0; i<10; i=i+1)
        tharray[i].Init("Customer")
        tharray[i].Fork(Customer,i)
        currentThread.Yield()
    endFor
  ThreadFinish()

endFunction

function Barber(p:int)
       while true
       customer.Down()
       lock.Lock()
       numwait=numwait-1
       PrintBarberStatus("start")
       hairCutFinished = false
       barber.Up()
       lock.Unlock()
       CutHair()
       endWhile
endFunction

function Customer(p:int)
       PrintCustomerStatus('E',p)
        lock.Lock()
        if numwait < CHAIRS
           numwait = numwait +1
           PrintCustomerStatus('S',p)
           --PrintChairStatus()
           customer.Up()
           lock.Unlock()
           barber.Down()
           GetHairCut(p)
       else
           lock.Unlock()
       endIf
      PrintCustomerStatus('L',p)
 endFunction

function CutHair()
      --var
        --i:int        --PrintBarberStatus("start")
        while (hairCutFinished == false)
          --currentThread.Yield()
        endWhile
        PrintBarberStatus("end")
endFunction

 function GetHairCut(p:int)
```

```
        var
            i:int
         PrintCustomerStatus('B',p)
          for i=0 to 1000                             ---Different Test cases with
diffrent delay
   endFor                                      ---Test case1&2: Dealy:100 Test
case3:Delay-10 Testcase4:Delay-1000
          currentThread.Yield()
          for i=0 to 1000
          endFor
        PrintCustomerStatus('F',p)
        hairCutFinished = true
 endFunction

function PrintBarberStatus(status:ptr to array of char)
          printLock.Lock()
          printChar ('\n')
          PrintChairStatus()
          print("  ")
          print(status)
          print(" ")
          printLock.Unlock()
endFunction

function PrintCustomerStatus(status:char,custNum:int)
       var
           i:int
        printLock.Lock()
        printChar ('\n')
        PrintChairStatus()  -- chair status
        print("          ")    -- barber status buffer
        for (i=0;i<custNum;i=i+1)
        print("   ")
        endFor
        printChar(status)
        printLock.Unlock()
endFunction

function PrintChairStatus()
     if(numwait==0)
      print("-----")
      --print("     ")
     endIf
     if(numwait==1)
      print("X----")
      --print("     ")
     endIf
     if(numwait==2)
      print("XX---")
      --print("     ")
     endIf
     if(numwait==3)
      print("XXX--")
      --print("     ")
     endIf
    if(numwait==4)
      print("XXXX-")
      --print("     ")
     endIf
     if(numwait==5)
      print("XXXXX")
      --print("     ")
     endIf
endFunction

endCode
```