U18CO018

Shekhaliya Shubham

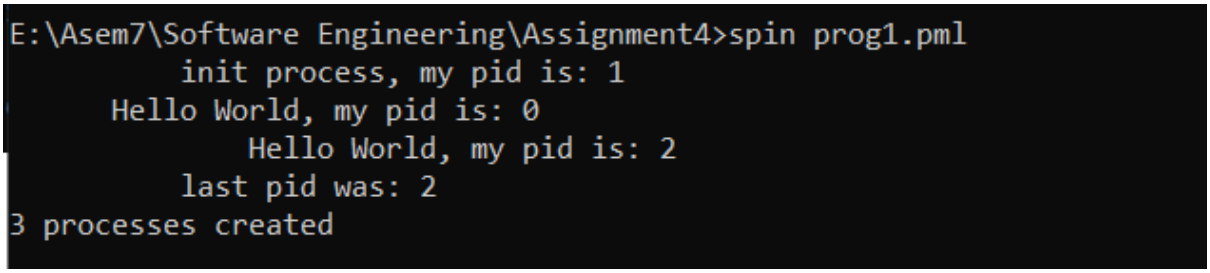
Software engineering

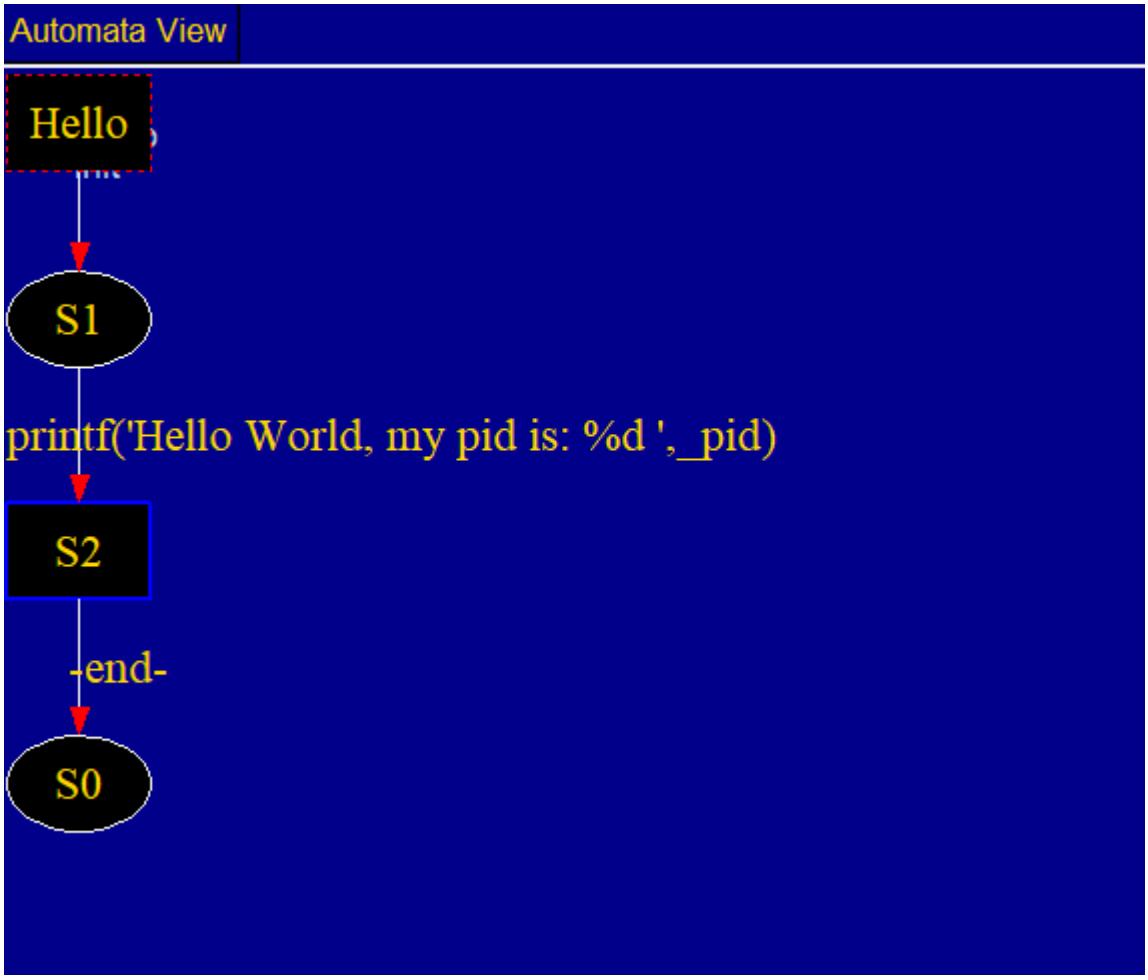
Lab Assignment 4

1. Write a program to create a process that prints “Hello World”. Use run in init process to instantiate it and _pid to print the ids of all create processes.

```
active proctype Hello() {
   printf("Hello World, my pid is: %d\n", _pid);
}

init {
   int lastpid;
   printf("init process, my pid is: %d\n", _pid);
   lastpid = run Hello();
   printf("last pid was: %d\n", lastpid);
}
```



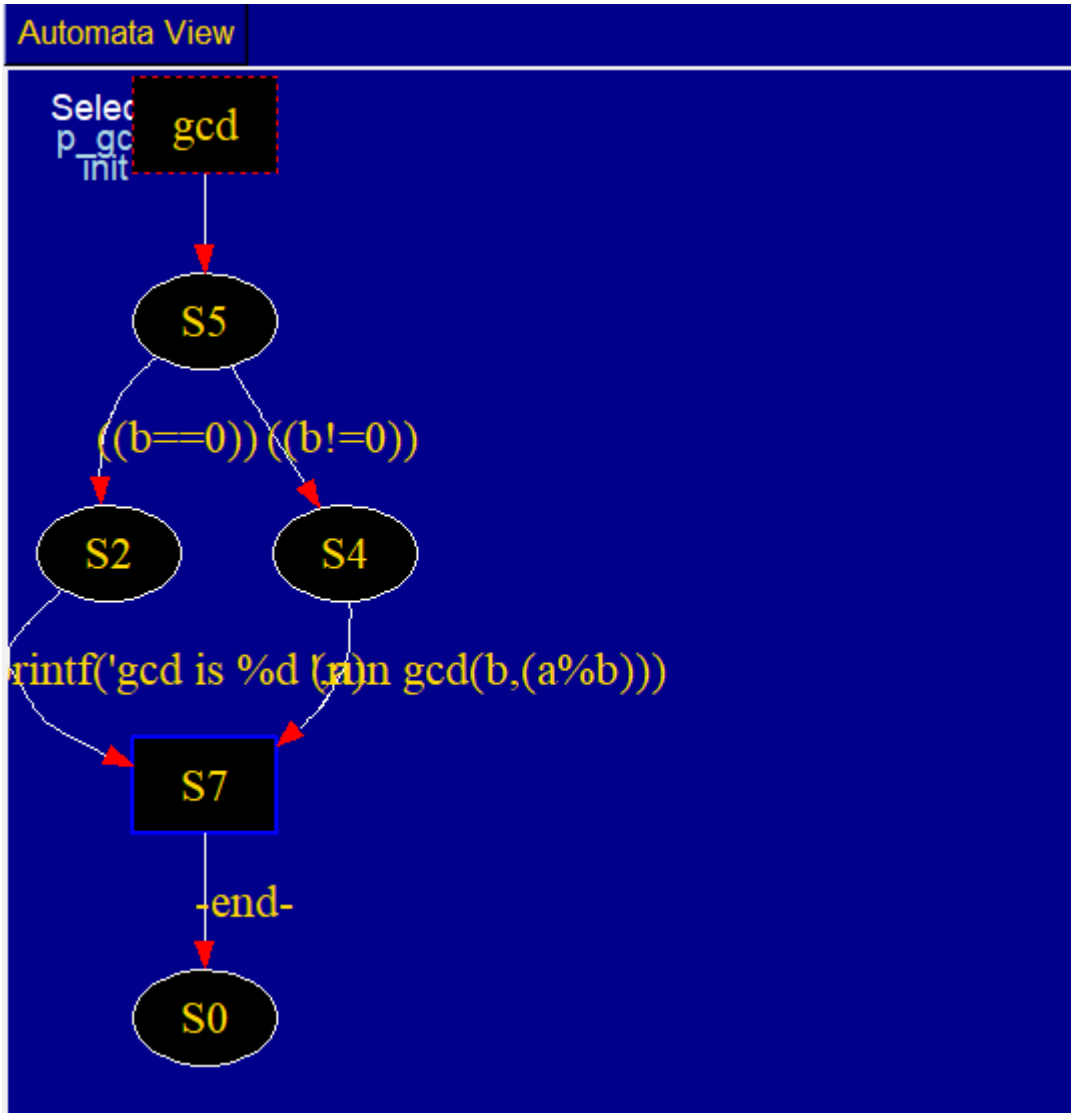

Automata View
Hello
S1
printf('Hello World, my pid is: %d ',_pid)
S2
-end-
S0

2. Model Euclid's algorithm for Greatest Common Divisor.

```
proctype gcd(int a;int b) {
   if
   :: (b == 0) -> printf("gcd is %d\n", a)
   :: (b != 0) -> run gcd(b, a % b)
   fi
}

init {
        printf("gcd for 14 and 21\n");
   run gcd(14, 21);
}
```

```
E:\Asem7\Software Engineering\Assignment4>spin prog2.pml
      gcd for 14 and 21
                     gcd is 7
5 processes created
```


Automata View
Selec
p_gc
init
gcd
S5
((b==0))
((b!=0))
S2
S4
rintf('gcd is %d (,a)n gcd(b,(a%b)))
S7
-end-
S0

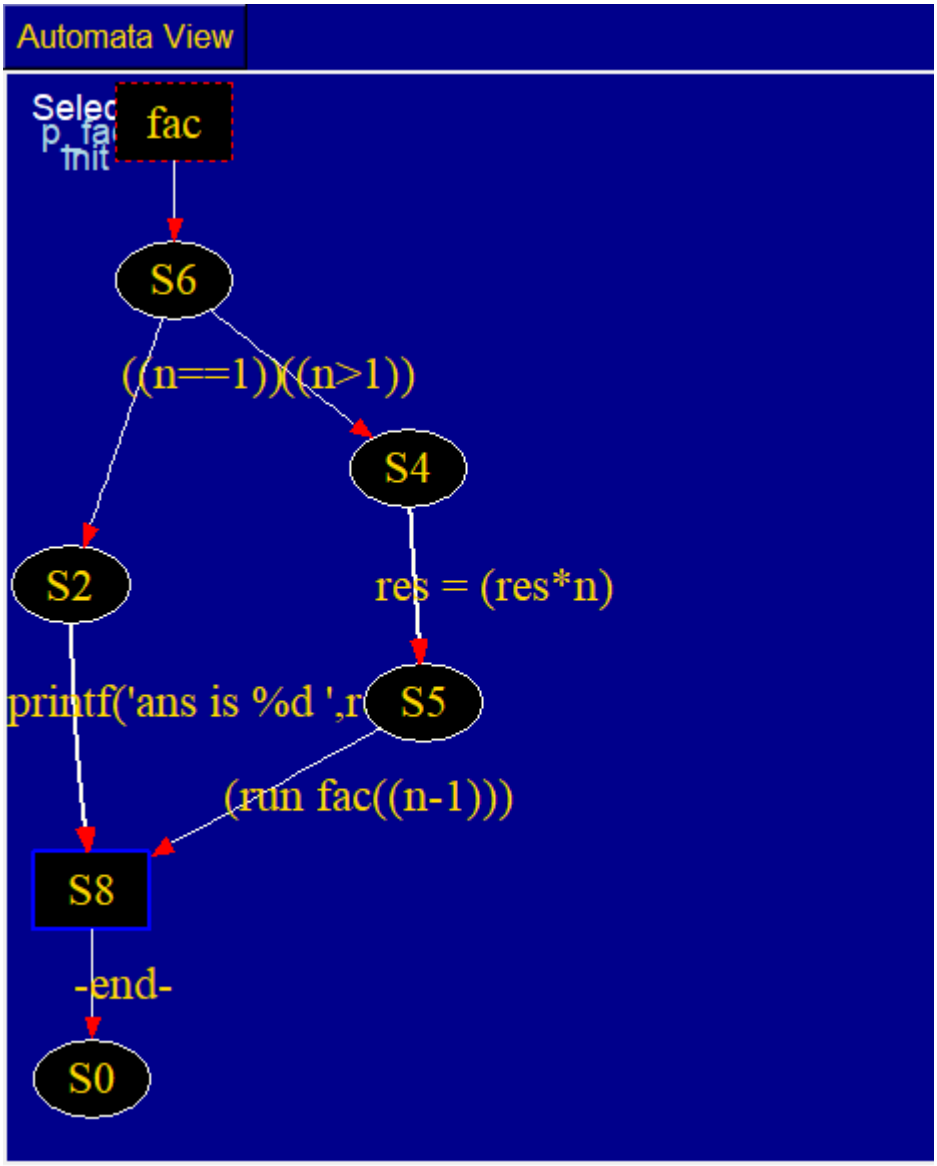
3. Create a process factorial(n, c) that recursively computes the factorial of a given non-negative integer “n”.

```
int res = 1;


proctype fac(int n) {
   if
   :: (n == 1) -> printf("ans is %d\n", res)
   :: (n > 1 ) -> res = res * n; run fac(n-1)
   fi
}


init {
        printf("factorial of 5 is\n");
   run fac(5);
}
```

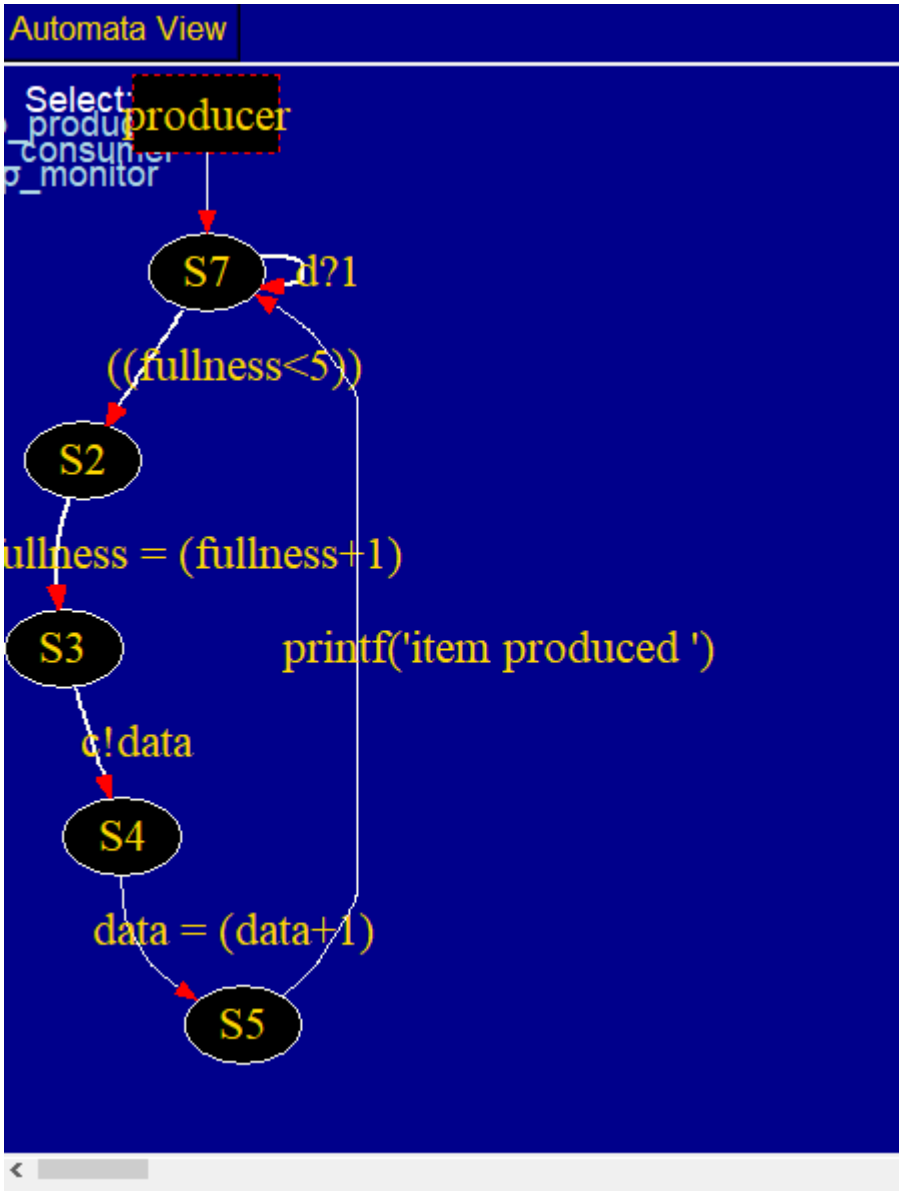
```
E:\Asem7\Software Engineering\Assignment4>spin prog3.pml
      factorial of 5 is
                          ans is 120
6 processes created
```

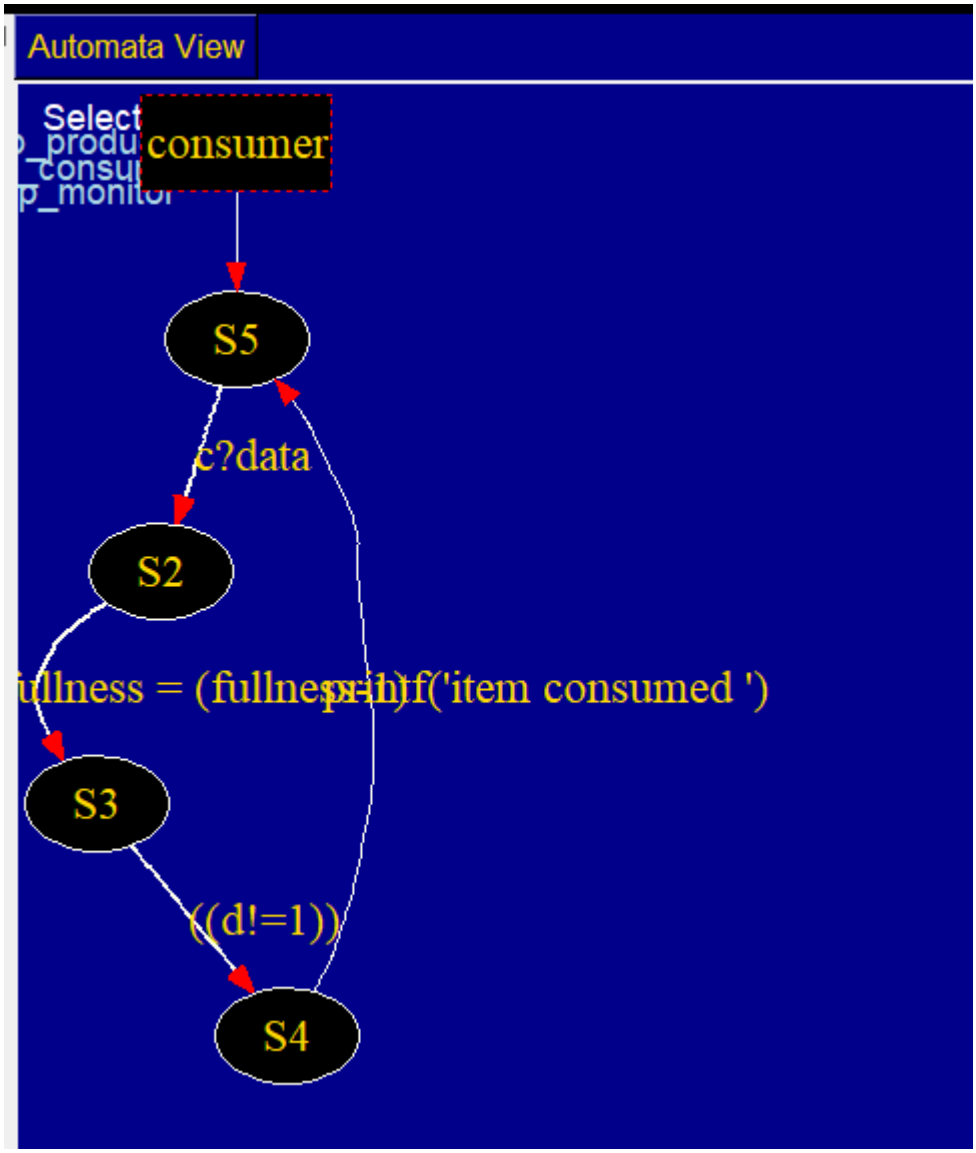

Automata View
fac
S6
((n==1))((n>1))
S4
S2
res = (res*n)
printf('ans is %d ',r
S5
(rtn fac((n-1)))
S8
-end-
S0

4. Create a Promela model for producer-consumer problem with buffer size 5.

```
#define SIZE 5
chan c = [6] of {byte};
chan d = [true] of {bool};
byte fullness = 0;
active proctype producer() {
   byte data;
   do
   :: fullness < SIZE -> fullness = fullness + 1;
                        c ! data;
                        data++;
                        printf("item produced\n")
   :: d ? true;
   od
}
active proctype consumer() {
   byte data;
   do
   :: c?data; fullness = fullness - 1; d != true; printf("item consumed\n")
   od
}
active proctype monitor() {
   assert (fullness <= SIZE);
}
```

```
E:\Asem7\Software Engineering\Assignment4>spin prog4.pml
      item produced
      item produced
          item consumed
      item produced
          item consumed
          item consumed
      item produced
          item consumed
      item produced
          item consumed
      item produced
          item consumed
      item produced
          item consumed
          item consumed
      item produced
      item produced
          item consumed
      item produced
          item consumed
      item produced
          item consumed
      item produced
          item consumed
          item consumed
```


Automata View
Select:
producer
g_monitor
S7
d?1
((fullness<5))
S2
ullness = (fullness+1)
S3
printf('item produced ')
c!data
S4
data = (data+1)
S5


Automata View
consumer
S5
c?data
S2
printf('item consumed ')
S3
((d!=1))
S4