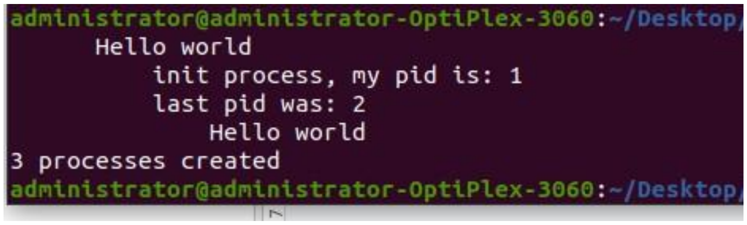
SE LAB ASSIGNMENT 4

U19CS082
SOURABH PATEL

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 created processes.

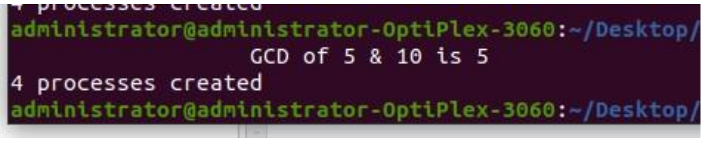
```
active proctype Hello(){
printf("Hello world\n")
}
init{
int lastpid;
printf("init process, my pid is: %d\n",_pid);
lastpid = run Hello();
printf("last pid was: %d\n", lastpid);
}
```



```
administrator@administrator-OptiPlex-3060:~/Desktop/
      Hello world
          init process, my pid is: 1
          last pid was: 2
              Hello world
3 processes created
administrator@administrator-OptiPlex-3060:~/Desktop/
```

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

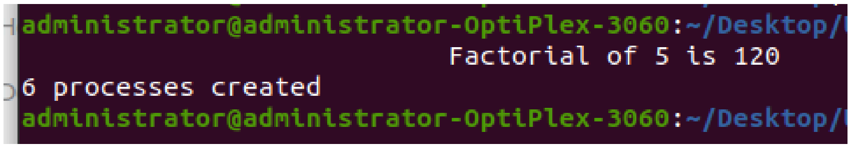
```
proctype gcd(int a; int b){
    if
    :: (b == 0) -> printf("GCD of 5 & 10 is %d\n", a)
    :: (b != 0) -> run gcd(b, a%b)
    fi
}
init{
    run gcd(5,10);
}
```



```
4 processes created
administrator@administrator-OptiPlex-3060:~/Desktop/
                  GCD of 5 & 10 is 5
4 processes created
administrator@administrator-OptiPlex-3060:~/Desktop/
```

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(" Factorial of 5 is %d\n",res)
    :: (n >= 2) -> res = res *n; run fac(n-1)
    fi
}
init{
    run fac(5);
}
```

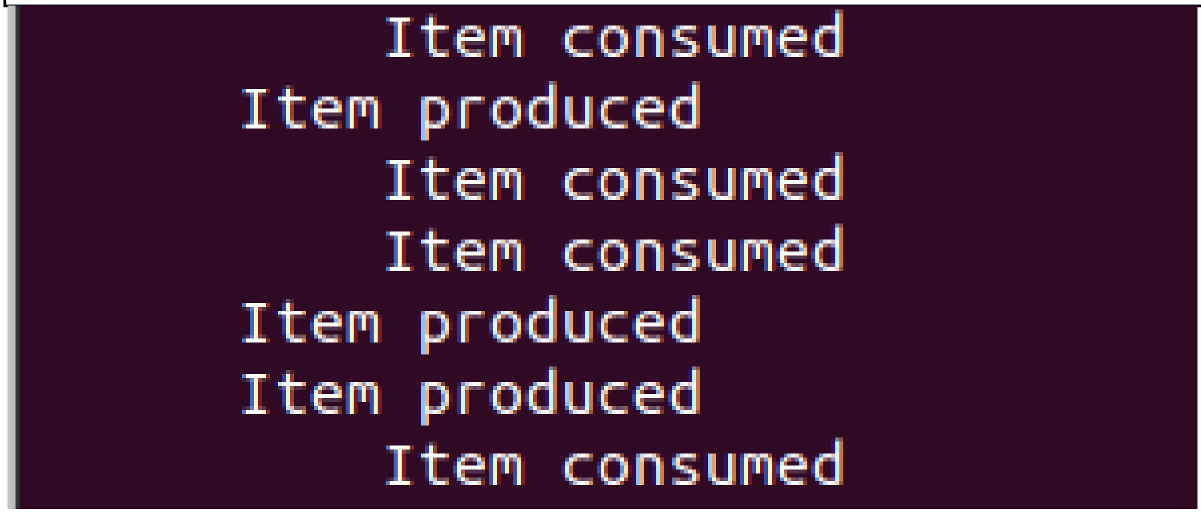


```
administrator@administrator-OptiPlex-3060:~/Desktop/
                          Factorial of 5 is 120
6 processes created
administrator@administrator-OptiPlex-3060:~/Desktop/
```

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 );
}
```



```
          Item consumed
      Item produced
          Item consumed
          Item consumed
      Item produced
      Item produced
          Item consumed
```