1.6

Análisis de complejidad temporal del 1.5

```
if [ $# -eq 1 ]; then
    cont=1
    for i in $(seq 1 $1); do
        cont=$(($cont*$i))
    done
    n_fact=$((cont))
    echo $n_fact
    else
        echo -n "debe ingresar solo un parametro: "
        read n
        (./Shell_1_factorial.sh $n)
fi
```

Shell_1_factoriales.sh

Primero hemos de mirar la complejidad del programa del inciso 1.1, dado que lo llamaremos a posteriori. Nótese que la única iteración del programa es un for que recorre los números desde el 1 hasta el parámetro dado. Dado que el for se encuentra en un if que carece de elif, entonces podemos que su complejidad temporal es: O(n+1)=O(n).

```
n=20
r=3
rest=$(($n-$r))

n_fact=$(./Shell_1_factorial.sh $n)
rest_fact=$(./Shell_1_factorial.sh $rest)

var=$(($n_fact / $rest_fact))

echo $var
```

Shell_5_variaciones.sh

Como es evidenciable, el programa del inciso 1.5 hace dos llamados a *Shell_1_factorial.sh*, con lo que podemos decir que su complejidad temporal es equivalente a la del inciso 1.1. (O(N)). Ahora bien, dado que los llamados se hacen de forma independiente podemos decir que la complejidad temporal del ejercicio 1.5 está dada por O(N)+O(N)=2O(N)=O(N).