Ejercicio 1

Dada la siguiente especificación y el siguiente programa, responda:

Especificación:

Pre: 1) Tablero 1 x 5 (filas x columnas).
2) El cabezal está en el extremo SO del tablero.
3) En el extremo SO hay dos bolitas rojas y no hay bolitas en ninguna otra celda.

Post: 1) El tablero sólo tiene bolitas rojas.
2) Mirando el tablero del Oeste al Este cada celda tiene el doble de bolitas que la anterior.

```
program{

        2BolitasRojas()

        Mover(Este)

        4BolitasRojas()

        Mover(Este)

        4BolitasRojas()
        4BolitasRojas()

        Mover(Este)

        16BolitasRojas()

        Mover(Este)

        16BolitasRojas()
        16BolitasRojas()
}

procedure 2BolitasRojas{
        Poner(Rojo)
        Poner(Rojo)
}

procedure 4BolitasRojas{
        2BolitasRojas()
        2BolitasRojas()
}
```

```
procedure 16BolitasRojas{
        4BolitasRojas()
        4BolitasRojas()
        4BolitasRojas()
        4BolitasRojas()
}
```

a. Indicar que errores y malas prácticas de programación sigue esta propuesta de solución (marcarlas en el código).

b. Proponer modificaciones en el código para que este pueda funcionar y para que la solución sea lo más comprensible posible.

c. Justifique por qué su propuesta es superadora y qué ventajas tiene programar así

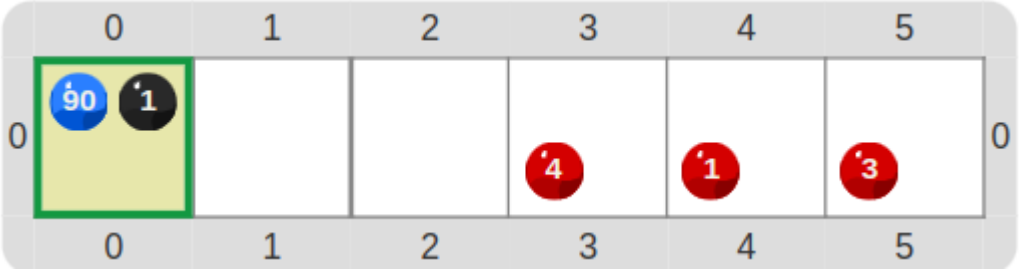
Ejercicio 2

¡Hay que apagar un incendio! Se extiende un incendio a lo largo del tablero. Hay un bombero con una reserva de agua que tiene que apagarlo celda por celda.
Las representaciones son:
1 bolita roja = 1 fuego
1 bolita azul = 1 agua
1 bolita negra = El bombero

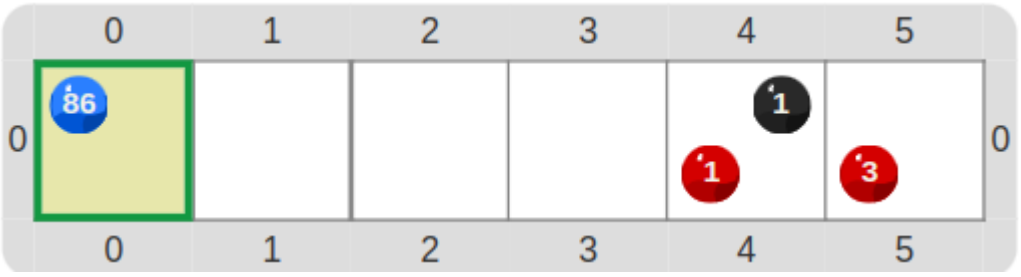
El bombero tiene que apagar los incendios uno por uno, es decir, no puede avanzar hasta que no haya apagado el incendio más cercano que tenga. Para apagar cada incendio, se apaga 1 unidad de fuego con 1 unidad de agua. Si se queda sin agua tiene que volver a buscar más. El bombero puede cargar la cantidad de agua que quiera. Para no gastar agua de más, debe calcular exactamente la cantidad que necesita. Debe repetir esto hasta apagar todas las celdas. Se asume que la capacidad de agua es muy grande y no se acaba.

A continuación se muestra un ejemplo de los estados del tablero:

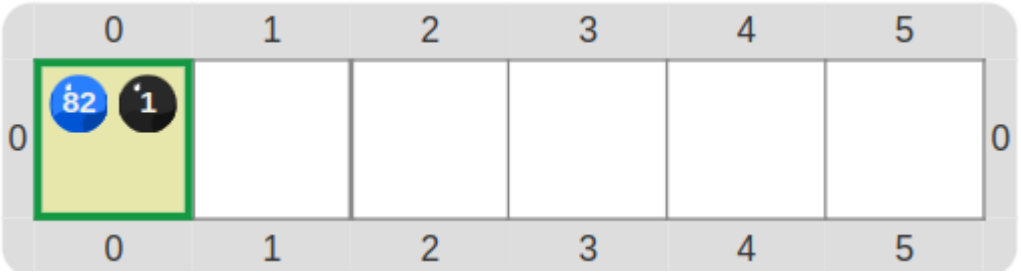
Tablero inicial:



Tablero intermedio:



Tablero final:



<u>Especificación:</u>

Pre: 1) Tablero 1 x n (1 fila, n columns), n > 5.
2) En el extremo Oeste hay 90 bolitas azules y una bolita negra.
3) A partir de la tercer columna hay una secuencia de celdas con sólo bolitas rojas en

distintas cantidades.
4) El cabezal está en el extremo Oeste del tablero.
5) No hay bolitas verdes en ninguna celda.

Post: 1) El tablero no tiene bolitas rojas.
2) La cantidad de bolitas azules en el extremo Oeste es 90 menos la cantidad total de fuego.
3) No hay bolitas verdes en ninguna celda.
4) El cabezal está en el extremo Oeste del tablero. Hay una bolita negra en el extremo Oeste.

a. Diseñar un programa en Gobstones que simula cómo el bombero apaga el incendio. Puede asumir que los siguientes procedimientos ya están implementados.

```
procedure MoverBomberoEnDireccion(direccion){
Pre: 1) La dirección solo puede ser Este u Oeste.
                2) En la celda donde se encuentra el cabezal debe haber una bolita negra.
Post: 1) El cabezal se encuentra en la celda contigua a su posición en el tablero
                    inicial.
                 2) En la posición original no está la bolita negra, en la posición contigua hay
una bolita negra.
}

procedure MoverBomberoConAguaEnDireccion(cantidad_agua, direccion){
 /*
        Pre: 1) La dirección solo puede ser Este u Oeste.
              2) En la celda donde se encuentra el cabezal debe haber una bolita negra.

        Post: 1) El cabezal se encuentra en la celda contigua a su posición en el tablero
                    inicial.
                 2) En la posición original no está la bolita negra y la cantidad de bolitas azules
es la que había menos cantidad_agua. En la nueva posición hay una bolita negra y la
cantidad de bolitas azules es cantidad_agua.
*/
}
```

Ejercicio 3

Indique si los siguientes enunciados son verdaderos o falsos. En caso de ser falso, justifique.

- La pre y la postcondición de un problema son predicados lógicos que describen el estado inicial y final antes y después de ejecutar un programa.
- La pre y la postcondición dan información acerca de cómo hay que resolver el problema.
- Para modelar un problema debe aclararse en qué lenguaje de programación se implementará su solución, ya que el modelado depende de la representación que se use.
- El uso de parámetros es útil cuando un procedimiento aparece repetido muchas veces para distintos valores. Permite crear un solo procedimiento general que recibe el parámetro como argumento.