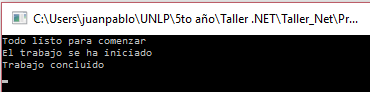
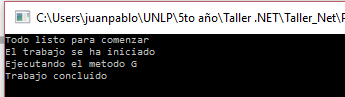
Practica 7

1. Salida por consola:

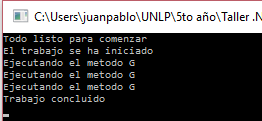


1. Borrando la línea “o.Trabajando = **new TrabajandoEventHandler”:**
   1. Se produce un error en el programa y este se interrumpe durante su ejecución.
   2. “El campo 'Ej1.Trabajador.Trabajando' nunca se asigna y siempre tendrá el valor predeterminado null”.
   3. Agregando el **if**(Trabajando != **null**) Trabajando(); se soluciona el problema.
2. Modificando el programa en las líneas definidas en el enunciado:
   1. Salida:

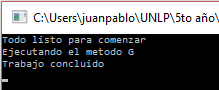


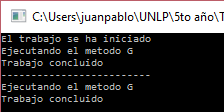
* 1. Se invocan siguiendo el orden en que fueron agregados al delegado (orden FIFO).

1. Modificando el programa en las líneas definidas en el enunciado:
   1. Que un delegado contenga más de una instancia del mismo método no produce ningún error, es una opción válida.
   2. Salida por consola:



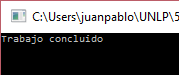
* 1. ¿?

1. Modificando el programa en las líneas definidas en el enunciado:
   1. Las siguientes líneas son equivalentes:
      1. o.Trabanado = o.Trabando + New TrabajandoEventHandler(G)
      2. .Trabajando += new TrabajandoEventHandler(G);
   2. Salida por consola ingresando la línea en el enunciado:
      1. o.Trabanado = New TrabajandoEventHandler(G)
      2. 
2. Modificando el programa en las líneas definidas en el enunciado:
   1. Salida por consola:

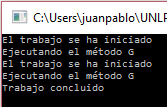
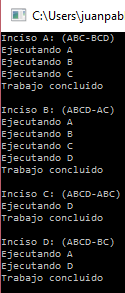
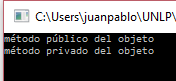
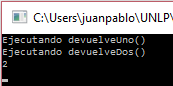
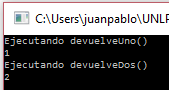
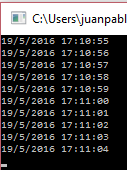


* 1. El operador “-=” borra el evento definido del eventHandler, en este caso el evento será el método F.
  2. Podria reemplazar la línea “o.Trabajando -= new TrabajandoEventHandler(F)” por: “o.Trabajando-=F”.
  3. Si el eventHandler posee varias instancias de un método, la operación de resta solo borrara 1 sola de dichas instancias.

1. Modificando el programa en las líneas definidas en el enunciado:
   1. Salida por consola:



* 1. El operador “-“ está sobrecargado.

1. Código:  
    delegado1 += **new** **TrabajandoEventHandler**(F);  
          delegado1 += **new** **TrabajandoEventHandler**(G);  
          delegado2 += **new** **TrabajandoEventHandler**(F);  
          delegado2 += **new** **TrabajandoEventHandler**(G);  
          Trabajador o=**new** **Trabajador**();  
          o.Trabajando = delegado1+delegado2;  
          o.**Trabajar**();  
     
   Salida por consola:  
     
   Por lo tanto delegado1+delegado2 = F+G + F+G
2. Salida por consola:  
   
3. Salida por consola:  
     
   Obsérvese que, cuando se realiza una llamada a través de un objeto delegado no se tienen en cuenta los modificadores de visibilidad de los métodos que se ejecutarán, lo que permite llamar desde un tipo a métodos privados de otros tipos que estén almacenados en un delegado.
4. Salida por consola:  
     
     
   delegado=**new** **tipoDelegado**(devuelveUno);  
   delegado+=**new** **tipoDelegado**(devuelveDos);  
   **int** i= **delegado**(); 🡪 Se le asigna el 2  
     
   Nota: Observe que en la instrucción “delegado=new tipoDelegado(devuelveUno);” no se ha utilizado el operador “+=” como veníamos haciendo en ejemplos anteriores. Esto se debe a que delegado se ha definido como una variable local en el método Main(), por lo tanto el compilador no permite expresiones de lectura sobre esta variable verificando que nunca fue asignada.
   1. Salida por consola:  
      
   2. Delegado.**GetInvocationList()** retorna un array con los delegados asignados a la variable que la llama. En el ejericio retorna un array de dimensión 2:  
      array[0] 🡪 devuelveUno  
      array[1] 🡪 devuelveDos
   3. Si.
   4. El programa cuenta 10 segundos y muestra fecha y hora por consola. Una vez finalizado los 10 segundos se detiene a través de un evento.
   5. Salida por consola:  
      
5. Correcciones:  
   - delegate void **TicEventHandler**(TicEvenArgs e);  
   - class TicEvenArgs:EventArgs  
       {  
        **public** DateTime horaActual{get;set;}  
       }  
   - TicEvenArgs e  
    e.horaActual = hora;  
     **Tic**(e);