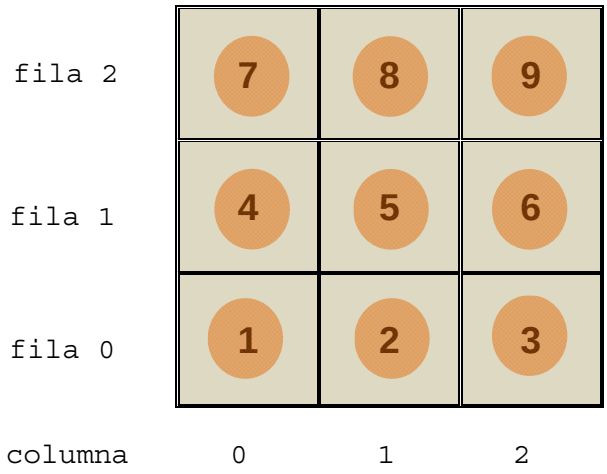
Práctica 1
Pasa la calculadora

Fecha de entrega: 7 de diciembre a las 23:55 horas

En *matematicasdivertidas.com* se propone el siguiente juego[1]:

> Dos jugadores **A** y **B** juegan de la manera siguiente: **A** enciende la calculadora y pulsa un dígito, y a continuación pulsa la tecla "**+**". Pasa la calculadora a **B**, que pulsa un dígito que ha de estar en la misma fila o columna que el último dígito pulsado por **A** y ser diferente de éste; a continuación pulsa "**+**" y le devuelve la calculadora a **A**, que repite la operación y así sucesivamente. Pierde el juego el primer jugador que alcanza o supera la suma 31.

La disposición de las teclas de los dígitos que se pueden pulsar es:



1. Descripción de la práctica

En esta práctica tienes que ir desarrollando de manera incremental un programa que permita jugar a *Pasa la calculadora* contra la máquina y llevar un informe sobre el uso del programa. En cada partida, el programa decidirá quién empieza (usuario o máquina) de forma aleatoria, y en cada turno leerá un dígito (en caso de jugar el ordenador, el dígito lo generará el propio programa), comprueba que cumple las condiciones de las reglas del juego, y muestra la suma acumulada. Si la suma alcanza o supera 31 el jugador que ha dado el último número habrá perdido.

[1] http://www.matematicasdivertidas.com/Juegos%20con%20Calculadora/juegos%20con%20calculadora.html#pasa2

Versión 1 (Pasa la calculadora, una partida)

En esta primera versión del programa sólo se permitirá jugar una partida. Después de un saludo pidiendo el nombre del usuario, el programa decide al azar quién empieza. Cuando le toque el turno al usuario el programa le mostrará el cuadrado con las opciones, le solicitará que introduzca un dígito, comprobará que el dígito introducido es correcto y mostrará por pantalla la suma. Si el usuario desea abandonar la partida deberá introducir el dígito 0. En el momento en que la suma llegue a 31 o la supere el programa comunica al usuario quién ha ganado y se despide. Ejemplo de ejecución:

```
¡Bienvenido a pasa la calculadora!
¿Cómo te llamas? Ana
Hola Ana
Empiezas tú
   7     8     9
   4     5     6
   1     2     3
Introduce un dígito (0 para abandonar): 2
Suma: 2
Yo pulso: 8
Suma: 10
   7     8     9
   4     5     6
   1     2     3
Introduce un dígito (0 para abandonar): 2
Suma: 12
Yo pulso: 8
Suma: 20
   7     8     9
   4     5     6
   1     2     3
Introduce un dígito (0 para abandonar): 8
Error: tiene que ser distinto de 8 y estar en la misma fila o columna
   7     8     9
   4     5     6
   1     2     3
Introduce un dígito (0 para abandonar): 4
Error: tiene que ser distinto de 8 y estar en la misma fila o columna
   7     8     9
   4     5     6
   1     2     3
Introduce un dígito (0 para abandonar): 9
Suma: 29
Yo pulso: 3
Suma: 32

¡Enhorabuena has ganado!
Hasta la próxima Ana (pulsa una tecla)
```

Define al menos la constante `META` con el valor 31 y el tipo enumerado `tJugador` para `Nadie`, `Automata` y `Persona`. Implementa al menos las siguientes funciones:

- ✓ `tJugador pasaCalculadora():` Conduce el desarrollo del juego y devuelve el ganador. Si se abandona devuelve `Nadie`. Utiliza, directa o indirectamente, las funciones que siguen.
- ✓ `tJugador quienEmpieza():` Decide aleatoriamente quién empieza (`Automata` o `Persona`).
- ✓ `bool mismaFila(int ultimo, int nuevo)`: Devuelve `true` si `nuevo` está en la misma fila que `ultimo`; en caso contrario devuelve `false`. Observa que la fila que ocupa un dígito, numeradas desde 0, corresponde con `(digito - 1) / 3`.
- ✓ `bool mismaColumna(int ultimo, int nuevo)`: Devuelve `true` si `nuevo` está en la misma columna que `ultimo`; en caso contrario devuelve `false`. Observa que la columna que ocupa un dígito corresponde con `(digito-1) % 3`, numeradas desde 0.
- ✓ `bool digitoValido(int ultimo, int nuevo):` Devuelve `true` si `nuevo` cumple las reglas del juego con respecto a `ultimo` (misma fila o columna, pero distinto número) ; en caso contrario devuelve `false`.
- ✓ `int digitoAleatorio():` Devuelve un dígito aleatorio. Más adelante se explica cómo se puede calcular un número aleatorio.
- ✓ `int digitoAutomata(int ultimo):` Devuelve un dígito aleatorio que cumpla las reglas del juego con respecto a `ultimo`. Por ejemplo, `digitoAutomata(2)` puede devolver `1,3,5` u `8`. Con esta función calculará la máquina su jugada. Hará uso de la función anterior.
- ✓ `int digitoPersona():`Pide un dígito al jugador. Sólo devolverá un valor válido (entre 0 y 9). Para cada valor no válido, mostrará un error y solicitará un nuevo dígito.
- ✓ `int digitoPersona(int ultimo):` Pide un dígito al jugador mostrando el teclado. Sólo devolverá un valor que cumpla las reglas del juego o 0. Para cada valor no válido, mostrará un error y solicitará un nuevo dígito. Hará uso de la función anterior.

Para generar números aleatorios utiliza las funciones `rand()` y `srand()` de la biblioteca `cstdlib`. Una secuencia de números aleatorios comienza en un primer número entero que se denomina semilla (*seed*). Una vez iniciada la secuencia, la función `rand()` genera, de forma pseudoaleatoria, otro entero positivo a partir del anterior. Si quieres que la secuencia esté compuesta por números en un determinado intervalo, tendrás que utilizar el operador `%`.

Para obtener un entero entre 1 y `M`: `(rand() % M) + 1`

Para obtener un entero entre 0 y `M`: `rand() % (M + 1)`

Lo que hace que la secuencia se comporte de forma aleatoria es la semilla. Una semilla habitual es el valor de la hora del sistema `time(NULL)`, de la biblioteca `ctime`, ya que es siempre distinta para cada ejecución.

Para establecer la semilla: `srand(time(NULL))`

Versión 2 (Pasa la calculadora con menú)

Añade un menú de opciones que permita elegir entre jugar una partida, mostrar la información acerca del programa, o salir del programa. El programa no terminará hasta que se seleccione la opción de salir. Ejemplo de ejecución:

```
¡Bienvenido a pasa la calculadora!
¿Cómo te llamas? Ana
Hola Ana
Selecciona una opción

1 - Jugar
2 - Acerca de
0 - Salir
Opción: 1

...

1 - Jugar
2 - Acerca de
0 - Salir
Opción: 0

Hasta la próxima Ana (pulsa una tecla)
```

Ahora el juego se comportará de la siguiente manera según la opción elegida:

1. Jugar: Se jugará una partida tal y cómo se hacía en la versión 1 de la práctica.
2. Acerca de: La información que se mostrará (créditos e instrucciones) será la que se encuentre en el archivo `versionPC.txt`.
3. Salir: El programa termina.

Ejemplo de ejecución usando la opción `Acerca de`:

```
1 - Jugar
2 - Acerca de
0 - Salir
Opción: 2

------------------------------------------------------
                Acerca de Pasa la calculadora

Práctica 1 Versión 3.1 (20/10/2014)

Fundamentos de Programación
Facultad de Informática
Universidad Complutense de Madrid

Autores:
     nombre y apellidos (grupo)
     nombre y apellidos (grupo)

Instrucciones: ...
------------------------------------------------------

1 - Jugar
2 - Acerca de
0 - Salir
Opción: _
```

Modifica el archivo `versionPC.txt` con los nombres de los autores e incluye unas sencillas instrucciones del juego.

Tendrás que añadir a la versión 1 al menos las siguientes funciones:

- ✓ `int menu()`: Muestra el menú, pide la opción y la devuelve como resultado. Sólo devolverá una opción válida. Para cada valor no válido, mostrará un error.
- ✓ `bool mostrar(string nombArch)`: Muestra en la consola el contenido del archivo de texto `nombArch`. Si el archivo no se encuentra, devuelve `false`, en otro caso `true`. Recuerda que el archivo debe terminar con un centinela (en este caso `XXX`) para que el programa sepa cuando parar de leer el fichero.

Versión 3 (Pasa la calculadora con menú e informe)

Se quiere tener información sobre el uso del programa, para lo cual se guardará en el archivo `informePC.txt` los siguientes datos:

```
Número de veces que se ha utilizado el programa
Número total de partidas jugadas (incluidos los abandonos)
Número total de partidas ganadas por el programa
Número total de abandonos
```

cada uno en una línea, y en el orden indicado. Ejemplo de archivo `informePC.txt`:

```
8
4
0
2
```

Para mantener el informe, al terminar el programa se actualizará el archivo sumando a los datos que ya había los datos de la última ejecución.

Añade la función:

- ✓ `bool actInforme(int jugadas, int ganadas, int abandonos):` Actualiza el archivo `informePC.txt` con los tres argumentos. En caso de no encontrar el archivo, lo crea y devuelve `false`; en otro caso devuelve `true`.

Versión 4 (Opcional)

Se observa, por los datos del informe, que el programa gana pocas veces, y se decide que hay que mejorar el comportamiento del jugador automático.

Haz a tu programa más inteligente mejorando la función `digitoAutomata()`. Además, añade al menú la opción de seleccionar el nivel de juego. Se podrá elegir el nivel 1 (el programa juega con la función `digitoAutomata()` descrita en la versión 1) o el nivel 2 (el programa juega con una función de `digitoAutomata()` mejorada, más inteligente).

2. Requisitos de implementación

No olvides incluir los prototipos de tus funciones. No utilices variables globales: cada función, además de los parámetros con los que se le pasa información en las llamadas, debe declarar localmente las variables que requiera.

El programa no debe utilizar arrays, archivos (salvo `versionPC.txt` e `informePC.txt`), o instrucciones de salto no estructuradas como `exit`, `break` (salvo en las cláusulas de la instrucción `switch`) y `return` (salvo en la última instrucción de las funciones que devuelven un valor).

3. Entrega de la práctica

La práctica se entregará en el Campus Virtual por medio de la tarea **Entrega de la Práctica 1**, que permitirá subir un archivo comprimido .zip con el código fuente de la última versión (`practica1.cpp`) y el archivo `versionPC.txt`. Asegúrate de poner el nombre de los miembros del grupo en un comentario al principio del archivo de código fuente.

El fichero subido deberá tener el siguiente nombre: Practica1GrupoXX.zip, siendo XX el número de grupo. Uno sólo de los miembros del grupo (no los dos) será el encargado de subir la práctica.

La práctica debe funcionar usando Visual Studio, que es la herramienta vista en clase. La práctica subida debe compilar y cumplir con la funcionalidad descrita en el enunciado.

Fecha límite de entrega: **7 de diciembre**