Práctica 1
¡Pasa la calculadora!

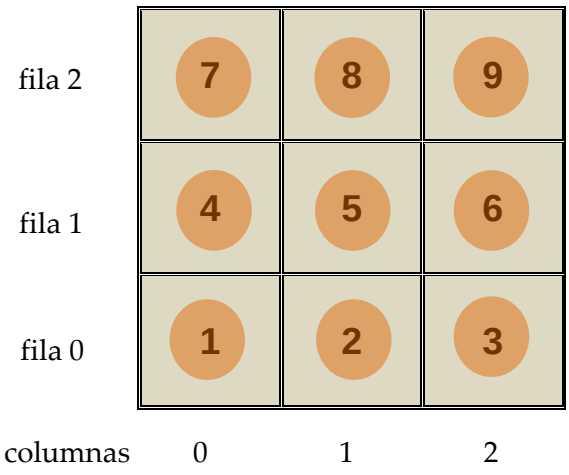
Fecha de entrega: 7 de diciembre

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. ¿Sabrías decir cuál de los dos jugadores tiene ante él una estrategia ganadora y en qué consiste?

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



[1] Tomado de: http://www.matematicasdivertidas.com/Juegos_con_Calculadora/juegos_con_calculadora.html#pasa2

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 el propio ordenador y llevar un informe sobre el uso del programa. En cada partida, el ordenador decidirá quién empieza (y será por tanto el jugador A) de forma *aleatoria*, y en cada turno se leerá un dígito (en caso de jugar el ordenador, el dígito lo generará el propio programa), se comprobará que cumple las condiciones que exige el juego, y se mostrará por pantalla la suma acumulada. Si esta alcanza o supera 31, el último jugador que ha introducido un número habrá perdido.

Versión 1 (¡Pasa la calculadora!, una partida)

Al inicio de la partida el programa decide al azar quien empieza. El dígito 0 se utilizará para indicar que se desea abandonar la partida. 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 como mínimo una 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 quien empieza.
- ✓ `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 (dígito - 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, numeradas desde 0, corresponde con (dígito-1)% 3.
- ✓ `bool digitoValido(int ultimo, int nuevo):` Devuelve `true` si `nuevo` cumple las reglas del juego con respecto a `ultimo`; en caso contrario devuelve `false`.
- ✓ `int digitoAleatorio():` Devuelve un dígito.
- ✓ `int digitoAutomata(int ultimo):` Devuelve un dígito que cumpla las reglas del juego con respecto a `ultimo`. Por ejemplo `digitoAutomata(2)` puede devolver `1, 3, 5` u `8`.
- ✓ `int digitoPersona():` Pide un dígito al jugador. Sólo devolverá un valor válido (entre 0 y 9). Para cada valor no válido, mostrará un error.
- ✓ `int digitoPersona(int ultimo):` Pide un dígito al jugador mostrando el teclado. Sólo devolverá un valor que cumpla las reglas del juego ó 0. Para cada valor no válido, mostrará un error.

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 (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 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)
```

Añade una función para el menú:

- ✓ `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.

`2- Acerca de ¡Pasa la calculadora!`

La información que se mostrará (créditos e instrucciones) será la que se encuentre en el fichero `versionPC.txt`.

Añade la función:

- ✓ `bool mostrar(string nombArch)`: Muestra en la consola el contenido del fichero de texto `nombArch`. Si el fichero no se encuentra, devuelve false, en otro caso true.

Ejemplo de ejecución:

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

Versión 3 (Informe)

Se quiere tener información sobre el uso del programa, para lo cual se guardará en el fichero `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.

Para mantener el informe, al terminar el programa se actualizará el fichero.

Añade la función:

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

Versión 4 (Opcional)

Haz tu programa más inteligente, mejora la función `digitoAutomata()` o añade otra y la opción de seleccionar el nivel de dificultad del juego, presentándolo como un submenú de la opción 1.

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, ficheros (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 antes de la fecha de entrega en el espacio online de la asignatura dentro del Campus Virtual. La entrega se realizará a través de la tarea **Entrega de la Práctica 1**, que permitirá subir un fichero con el código fuente de la última versión (`practica1.cpp`) y el fichero `versionPC.txt`. Asegúrate de poner todos los datos de la asignatura y de los miembros del grupo en los comentarios que van al inicio del fichero de código fuente.