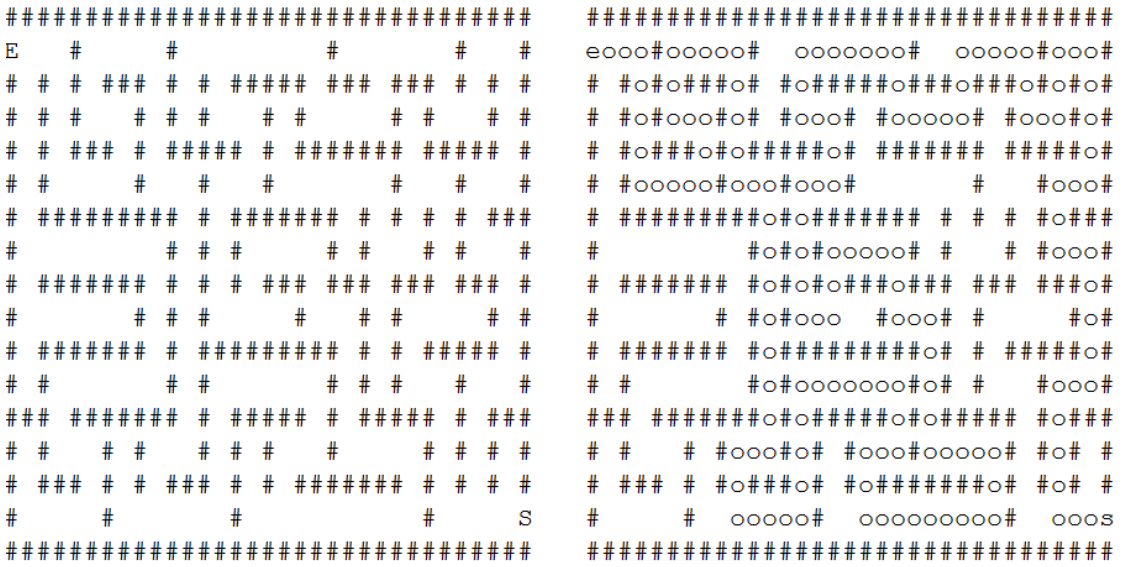
Laberinto (Recursión, backtracking)

En el archivo **laberinto.txt** encontrará una representación ASCII de un laberinto como el que se muestra en la figura de la izquierda. La celda marcada con una **E** es la entrada, **S** es la salida, **#** representan paredes infranqueables y los espacios son de libre tránsito. Se solicita encontrar el camino que lleva de la entrada a la salida del laberinto. Márquelo en el mapa con caracteres **o,** e imprímalo, como se muestra en la figura de la derecha.



Caminata al azar sin cruces

Una caminata al azar sin cruces es un proceso en el cual un caminante se desplaza como en una caminata al azar normal pero en cuanto vuelve a visitar un sitio previamente visitado, se detiene y deja de caminar. Este modelo es un paradigma para el estudio de las configuraciones espaciales de cadenas poliméricas y proteicas.
Cada caminante se ubica inicialmente en el centro de una red cuadrada (dimensión NxN), y en cada paso de tiempo elige aleatoriamente moverse a uno de los cuatro sitios vecinos, ubicados respectivamente, arriba, abajo, derecha o izquierda (si el caminante se sale de la grilla aparece por el borde inverso, ejemplo: si se va del borde derecho->aparece por el borde izquierdo). El caminante irá marcando los puntos visitados. Si en algún momento, el sitio elegido para moverse ya ha sido previamente visitado, la caminata se corta. Si luego de 10000 pasos la caminata aún sigue activa, forzar su interrupción.
En este problema se pide que se consideren 1000 caminatas al azar sin cruces y se almacene el largo de cada caminata y finalmente se analice la distribución de la longitud de las caminatas a través de un histograma de 20 intervalos.

Evitando el megabochazo

Un estudiante debe rendir examen en tres asignaturas (**A**, **B** y **C**). Su familia le ha exigido que, para irse de vacaciones, apruebe al menos una de ellas. Le quedan cuatro días para estudiar y no le da buen resultado estudiar más de una asignatura diferente el mismo día.

El centro de estudiantes, luego de años y años de estadística, dispone de los datos de la probabilidad $\mathbf{p_k(t)}$ de **reprobar** la asignatura **k** si se dedican **t** días de estudio a esa asignatura y entregan dichos datos en forma un archivo de texto a dos columnas, la primer columna son los días de estudio y la segunda la probabilidad de reprobar habiendo estudiado esa cantidad de días (que obviamente disminuye al aumentar los días de estudio...).

Nuestro estudiante consigue la información que necesita en los archivos A.txt, B.txt y C.txt y ahora necesita determinar cuántos de los **4 días** de que dispone debe dedicarle al estudio de cada una de las asignaturas, de forma de **minimizar** la probabilidad de reprobar las 3 asignaturas. Dicha probabilidad se calcula como $P_{R3} = p_A(t_A) \times p_B(t_B) \times p_C(t_C)$**,** siendo t_A**,** t_B ***y*** t_C el número de días dedicado a las asignaturas ***A, B y C*** *respectivamente* y por lo tanto deben cumplir que $t_A + t_B + t_C = 4$**.**

Escriba un programa que lea los datos de los 3 archivos y determine la combinación de días t_A**,** t_B ***y*** t_C que minimizan la probabilidad de reprobar las 3 asignaturas.

El problema del caballo

Este problema consiste en posicionar un caballo en un casillero de un tablero de ajedrez y realizando movimientos típicos de esa pieza, intentar visitar todas las casillas, sin repetir ninguna. Marcar cada casilla con su número de movida (1,2,3,...,64) e imprimir el tablero.

En campaña

La secretaria que tomó el dictado del discurso inaugural de campaña de nuestro benemérito candidato, contenido en el archivo llamado **discurso.txt**, ha cometido varios errores. En primer lugar ha colocado múltiples espacios donde sólo se requería uno, y en segundo lugar, en algunos lugares ha repetido algunas palabras (como “de de” o “la la”). Dado que se trata de una “amiga” personal del candidato, nos vemos imposibilitados de marcarle dichos errores y mucho menos de solicitarle que los arregle. Y como tampoco podemos despedirla, es de esperar que siga cometiendo esta clase de errores en sucesivos discursos. Por lo tanto, le solicitamos a Ud. que por el bien de nuestra causa, implemente dos funciones, que arreglarán estos problemas. Dichas funciones deberán respetar los siguientes prototipos:

a) **void elimina_espacios_redundantes(string & discurso);** // Esta función eliminará los espacios extras contenidos en el string que recibe como argumento.

b) **void elimina_palabras_repetidas(string & discurso);** //Esta elimina las palabras repetidas, dejando sólo una de ellas (“el el” => “el”) en el texto que se le pasa como argumento (y esperemos que a nuestro candidato no se le ocurra viajar a Bora Bora!!).

Tenés Empanadas Graciela

En un juego de mesa llamado Risk (o **TEG**) el resultado de las batallas se determina arrojando dados. En el caso de que haya muchos ejércitos de ambos bandos, el atacante arroja siempre 3 dados y el defensor puede elegir entre lanzar 1 o 2 dados. Una vez arrojados, los dados de ambos contendientes se ordenan de mayor a menor puntuación y se comparan 1 a 1, como muestra el ejemplo de la figura (rojo atacante y blanco defensor que decidió usar 2 dados).

Si el dado del defensor tiene un puntaje **mayor o igual** que el del atacante, el atacante pierde un ejército, sino el defensor pierde un ejército (en el ejemplo de la figura el defensor perdió dos ejércitos). Un cálculo de probabilidades dice que si el defensor tira con 1 solo dado, hay un 34% de chances de que el atacante pierda un ejército (y 66% de que el defensor lo pierda). En cambio, si el defensor tira 2 dados, hay un 34% de chances de que el atacante pierda 2 ejércitos, un 37% de que el defensor pierda 2 ejércitos y un 29% de que cada uno pierda 1 ejército. Así que a primera vista, pareciera que siempre le conviene al defensor tirar con 2 dados. Sin embargo, la elección del defensor se realiza una vez que el atacante ya tiró sus dados. Así que la pregunta es, teniendo a la vista los 3 dados del atacante, ¿cuándo le conviene al defensor tirar 1 dado y cuándo 2? Un consejo muy difundido es que si el segundo dado marca 3 o menos, le conviene al defensor tirar con 2 dados, en caso contrario tirar con 1. Realice una simulación para determinar si este consejo es acertado o no. Para facilitar la tarea, le pedimos que implemente las siguientes funciones (respetar los prototipos):

1. **vector<int> tirardados(int nd);**
   Esta función recibe el número de dados **nd**. Tira al azar **nd** números del **1** al **6** y los pone en un vector<int>. Lo ***ordena de mayor a menor*** y lo retorna.
2. **void imprimirdados(vector<int> da,vector<int> dd);**
   Esta función recibe dos vectores **da** y **dd**, que representan los dados del atacante y del defensor respectivamente, e imprime los valores de ambos arreglos

encolumnados y lado a lado para facilitar la comparación visual de la escaramuza. Por ejemplo para 3 dados del atacante y 2 del defensor podría ser así:

```
A <=> D
4 <=> 3
3 <=> 2
3
```

3. **int escaramuza(vector<int> da, vector<int> dd);**
   Esta función calcula el resultado de una escaramuza (una tirada de dados), según las reglas del juego antes explicadas, retornando:

   2 => cuando el defensor pierde 2 ejércitos y el atacante 0 ejércitos

   1 => cuando el defensor pierde 1 ejército y el atacante 0 ejércitos

   0 => cuando ambos pierden 1 ejército

   -1 => cuando el atacante pierde 1 ejército y el defensor 0 ejércitos

   -2 => cuando el atacante pierde 2 ejércitos y el defensor 0 ejércitos

Implemente un programa que utilizando las funciones anteriores realice una simulación de **n** escaramuzas donde el atacante siempre utiliza 3 dados y calcule cuántos ejércitos pierden en promedio el atacante y el defensor para los casos:

A. El defensor utiliza siempre 1 dado.
B. El defensor utiliza siempre 2 dados.
C. El defensor utiliza 1 dado si el resultado del segundo dado del atacante es **> 3**, sino utiliza 2 dados.

Nota: para calcular el promedio de los ejércitos perdidos, note que el número de ejércitos involucrados en una escaramuza es: 1 si el defensor tira 1 dado y 2 si el defensor tira 2 dados. O sea que luego de **n** escaramuzas de 3 vs 1 dados, el total de ejércitos perdidos por ambos contendientes es **n**, pero si las escaramuzas son de 3 vs 2 dados, el total de ejércitos perdidos por ambos contendientes es **2n.**