1. En una competencia deportiva se ingresarán uno por uno los tiempos en segundos que realizaron distintos nadadores en una competencia. Solicite al usuario el ingreso de un número N de nadadores. Valide esta entrada de modo que el mínimo sea 1 nadador y el máximo 16. Para cada uno ingrese su tiempo. Genere las siguientes estadísticas:

   - Tiempo promedio
   - Mejor tiempo.
   - Peor tiempo.

   Algunos ejemplos de diálogo de este programa serían:

Listing 1: Ejemplo 1

```
Ingrese numero de nadadores:5

Ingrese tiempo del nadador:12.1
Ingrese tiempo del nadador:15.4
Ingrese tiempo del nadador:13.2
Ingrese tiempo del nadador:11.5
Ingrese tiempo del nadador:14.2

El mayor tiempo fue: 15.4 segundos
El menor tiempo fue: 11.5 segundos
El tiempo promedio es: 13.28 segundos
```

Listing 2: Ejemplo 2

```
Ingrese numero de nadadores:3

Ingrese tiempo del nadador:11.2
Ingrese tiempo del nadador:14
Ingrese tiempo del nadador:9.8

El mayor tiempo fue: 14 segundos
El menor tiempo fue: 9.8 segundos
El tiempo promedio es: 11.6667 segundos
```

Listing 3: Ejemplo 3

```
Ingrese numero de nadadores:4

Ingrese tiempo del nadador:11.2
Ingrese tiempo del nadador:14.3
Ingrese tiempo del nadador:12.2
Ingrese tiempo del nadador:11.3
```

```
El mayor tiempo fue: 14.3 segundos
El menor tiempo fue: 11.2 segundos
El tiempo promedio es: 12.25 segundos
```

Listing 4: Ejemplo 4

```
Ingrese numero de nadadores:5

Ingrese tiempo del nadador:12.1
Ingrese tiempo del nadador:12.3
Ingrese tiempo del nadador:11.9
Ingrese tiempo del nadador:11.8
Ingrese tiempo del nadador:12.3

El mayor tiempo fue: 12.3 segundos
El menor tiempo fue: 11.8 segundos
El tiempo promedio es: 12.08 segundos
```

2. Elabore un programa que contenga una función llamada mayorAlNumero Esta función recibirácomoparámetroslossiguientesvalores:

   - Número a evaluar (numero): Es un número entero ingresado por el usuario
   - Primer número (n1): Es un número entero contra el que será comparado numero.
   - Segundo número (n2): Es un número entero contra el que será comparado numero.
   - Tercer número (n3): Es un número entero contra el que será comparado numero.

   La función retorna true si el numero es validado, es decir si es menor que n1, n2 y n3. La función retorna false en caso contrario. El programa principal incluirá un ciclo repetitivo con 5 iteraciones. En cada uno de las iteraciones, el usuario ingresará el número a validar y los tres números a ser evaluados por la función. Luego de realizar la validación a través de la función, muestra el resultado.

   Algunos ejemplos de diálogo de este programa serían:

Listing 5: Ejemplo 1

```
ITERACION 1
------------------
Ingrese numero a evaluar:56

Ingrese primer numero de prueba:122
Ingrese segundo numero de prueba:133
Ingrese tercer numero de prueba:144

El numero a evaluar es menor que los 3

ITERACION 2
------------------
Ingrese numero a evaluar:89

Ingrese primer numero de prueba:13
Ingrese segundo numero de prueba:14
Ingrese tercer numero de prueba:15

Este numero es mayor a alguno(s)
```

Listing 6: Ejemplo 2

```
ITERACION 1
------------------
Ingrese numero a evaluar:116

Ingrese primer numero de prueba:67
Ingrese segundo numero de prueba:87
Ingrese tercer numero de prueba:65
```

```
Este numero es mayor a alguno(s)

ITERACION 2
------------------
Ingrese numero a evaluar:45

Ingrese primer numero de prueba:100
Ingrese segundo numero de prueba:200
Ingrese tercer numero de prueba:400

El numero a evaluar es menor que los 3
```

Listing 7: Ejemplo 3

```
ITERACION 1
------------------
Ingrese numero a evaluar:1

Ingrese primer numero de prueba:34
Ingrese segundo numero de prueba:56
Ingrese tercer numero de prueba:78

El numero a evaluar es menor que los 3

ITERACION 2
------------------
Ingrese numero a evaluar:120

Ingrese primer numero de prueba:3}
Ingrese segundo numero de prueba:5
Ingrese tercer numero de prueba:6

Este numero es mayor a alguno(s)
```

3. Elabore un programa utilizando recursividad. El programa solicita el ingreso de un número múltiplo de 3, 5 o 7. Valide el ingreso. Elabore una función recursiva reduceMultiploquerecibecomoparámetros:

   - n: Numero entero ingresado por el usuario.
   - mult: mayor múltiplo posible entre 3, 5 o 7 para el numero ingresado por el usuario.

   La función retorna un entero: el número de veces que el mayor de los posibles enteros se encuentra contenido en n.

   Algunos ejemplos de diálogo de este programa serían:

Listing 8: Ejemplo 1

```
Ingrese un numero multiplo de 3, 5 o 7:36
Veces que se encuentra contenido el entero 3: 12
```

Listing 9: Ejemplo 2

```
Ingrese un numero multiplo de 3, 5 o 7:21
Veces que se encuentra contenido el entero 7: 3
```

Listing 10: Ejemplo 3

```
Ingrese un numero multiplo de 3, 5 o 7:80
Veces que se encuentra contenido el entero 5: 16
```

Listing 11: Ejemplo 4

```
Ingrese un numero multiplo de 3, 5 o 7:90
Veces que se encuentra contenido el entero 5: 18
```

1. El Ministerio de Transportes ha planteado una nueva forma de identificar alos vehículos, la nueva placa (patente) está formado por 6 dígitos: el primer dígito indicala procedencia (1=AMERICANO, 2=EUROPEO), los siguientes dos dígitos indican el año y últimos tres restantes es el correlativo. Por ejemplo, el código 217001 representa un vehículo Europeo del año 2017 y tiene el correlativo 001. Escriba un programa permita determinar de un total de 10 placas el porcentaje de vehículos con año de fabricación menores al 2015 y el total de vehículos Americano. Adicionalmente, se sabe que esta nueva forma solo aplica a vehículos fabricados después del año 2000.

   Input

   - Cada línea contiene un número entero que representa a las placas.

   Ouput

   - Imprime el % de vehículos fabricados antes del 2015 y el total de vehículos americanos

   Algunos ejemplos de este programa serían:

Listing 1: Ejemplo 1

```
Ingresar las placas: 216001 210001 114111 117000 213151
   212555 213555 118555 119999 202555
Output:
El % anterior al 2015 es: 60
El total de americanos es: 4
```

Listing 2: Ejemplo 2

```
Ingresar las placas: 219001 217001 114111 217000 213151
   213555 213000 218555 119999 102555
Output:
El % anterior al 2015 es: 50
El total de americanos es: 3
```

Listing 3: Ejemplo 3

```
Ingresar las placas: 119001 219001 114111 117000 213151
   213555 218000 218555 119999 102555
Output:
El % anterior al 2015 es: 40
El total de americanos es: 5
```

2. (8 points) Un grupo de Estudiantes están jugando a las cartas y cada uno extrae cartas del mazo, estas cartas esta representadas **PT** donde **P** es el poder y puede tomar los valores **(2,3,4,5,6,7,8,9,J,Q,K,A)** y **T** es el tipo que puede tomar los valores **(S,H,D,C)**. Cada estudiante está jugando con múltiples mazos y cada carta tiene un valor que se calcula multiplicando el poder y el tipo. Para ello, las cartas del 2 a 10 tienen el mismo poder que el valor y las cartas de **J a A** son **11 a 14** respectivamente. A los tipos se les asigna un multiplicador de la siguiente manera $(S->4, H->3, D->2, C->1)$. **Implemente un algoritmo, usando funciones en C++ y creando archivos .h y .cpp, que permita determinar el valor total que cada jugador tiene en su mano.**

Input

- La primera línea contiene un número entero $N(1 \leq N \leq 6)$, que es es la cantidad de cartas que tiene cada jugadores.
- Las siguientes líneas corresponde a las cartas separadas en el formato: **{PT, PT, PT, ... PT}** Donde **P(2,3,4,5,6,7,8,9,J,Q,K,A)** es el poder y **T (S,H,D,C)** es el tipo.

Ouput

- Se espera Imprimir el valor total que cada jugador tiene en su mano.

Algunos ejemplos de este programa serían:

Listing 4: Ejemplo 1

```
Ingresar la cantidad de cartas: 5
Ingresar cartas: 2C 4H 9H AS QS
Output: 145
```

Listing 5: Ejemplo 2

```
Ingresar la cantidad de cartas: 4
Ingresar cartas: QH QC QS QD
Output: 120
```

Listing 6: Ejemplo 3

```
Ingresar la cantidad de cartas: 6
Ingresar cartas: 3H 10S JC KD 5S 9S
Output: 102
```

Indicaciones:

- Considere al menos las siguientes funciones:
  - **getPoder**, que devuelva el poder de una carta,
  - **getvalor**, que devuelva el tipo de la carta
  - **calcularTotal**, que devuelva el valor total de la carta

- Considere usar la función substr de la librería $\#include < string >$ el cual permite extraer una subcadena, siendo la sintaxis lo siguiente: **Cadena.substr(expr1, expr2)**
  - **expr1**, específica la posición inicial de la sub cadena.
  - **expr1**, específica la cantidad de carácter a retornar.

3. Elabore un algoritmo que permita imprimir la figura de un triangulo usando **recursividad**

   Input

   - La única línea contiene un número entero N(1 N 10), que el tamaño de la gráfica.

   Ouput

   - Se espera Imprimir la figura del triangulo.

   Algunos ejemplos de este programa serían:

Listing 7: Ejemplo 1

```
Ingrese un n mero : 7
      *
     * *
    * * *
   * * * *
  * * * * *
 * * * * * *
* * * * * * *
```

Listing 8: Ejemplo 2

```
Ingrese un n mero : 6
      *
     * *
    * * *
   * * * *
  * * * * *
 * * * * * *
```

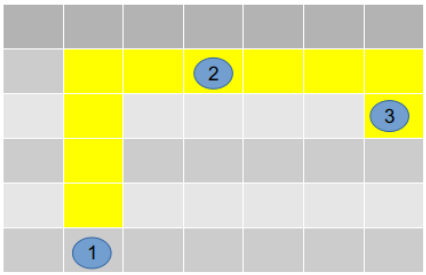
Listing 9: Ejemplo 3

```
Ingrese un n mero : 3
      *
     * *
    * * *
```

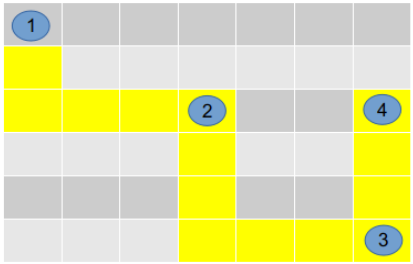
1. La métrica del taxista conocida también como distancia Manhattan permite medir la distancia del camino más corto entre dos puntos como la suma de las diferencias absolutas de sus coordenadas:

$$DM(P1, P2) = |P1.x - P2.x| + |P1.y - P2.y|$$

Como aplicación de dicha métrica se le pide diseñar un algoritmo para calcular la distancia total de recorrido a través k puntos para un tablero de NxM celdas. El recorrido empieza desde el primero punto hasta el ultimo en orden secuencial tal como se muestra en los siguientes ejemplos:



Celdas recorridas: 10



Celdas recorridas: 14

El programa debe contener:

- Solicitar un número válido de puntos $k >= 2$.
- Solicitar k puntos como posiciones en el plano cartesiano.
  - Se puede admitir valores decimales.
- Reportar la distancia total recorrida.

Algunos ejemplos de diálogo de este programa serían:

```
k: 0
k: 1
k: 3
P1: 2 1
P2: 4 5
P3: 7 4
Distancia total: 10
```

```
k: -1
k: 4
P1: 1 6
P2: 4 4
P3: 7 1
P4: 7 4
Distancia total: 14
```

2. Se le pide implementar un programa que grafique el porcentaje de avance laboral de un grupo de trabajadores. Hay un mínimo y máximo de tareas que deben realizar cada trabajador por día. Se le proporciona la función main y en base a ello implementar las funcionesrestantesenunheader.

   Maneje las siguientes constantes en el header:

```
const int MIN_TAREAS=5;          //minimo de tareas por dia
const int MAX_TAREAS=20;         //maximo de tareas por dia
const int MAX_TRABAJADORES=10;   //maximo de trabajadores
```

   La función príncipal es como sigue:

```
#include <iostream>
#include "funciones.h"
using namespace std;

int main() {
  srand(time(NULL));//semilla del random

  int total_tareas;
  int n = obtenerTrabajadores();
  cout<<"Graficando␣avance␣diario:\n";
  for(int i=0;i<n;i++){
    total_tareas = generarAvance();
    graficarAvance(total_tareas);
  }
  return 0;
}
```

   Implemente usted las siguientes funciones:

   - **obtenerTrabajadores()** solicita al usuario la cantidad de trabajadores a reportar. Debe validar que no excede el máximo permitido.
   - **generarAvance()** genera aleatoriamente una cantidad de tareas en el rango [MIN_TAREAS - MAX_TAREAS].
   - **graficarAvance(totaltareas)** grafica la barra porcentual, y al final de la barra coloca las tareas realizadas y su equivalente en procentaje de avance.

Algunos ejemplos de diálogo de este programa serían:

```
Cantidad de trabajadores:-5
Cantidad de trabajadores:0
Cantidad de trabajadores:15
Cantidad de trabajadores:10
Graficando avance diario:
||||||||||||||||||||||||||||||                                  |12:60%
||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||20:100%
|||||||||||||||||||||||||||||||||||||||||                        |17:85%
||||||||||||||||||||||||||||||||                                |13:65%
|||||||||||||||||                                               |7:35%
|||||||||||||||||                                               |7:35%
||||||||||||||||||||||||||                                      |11:55%
|||||||||||||||||                                               |7:35%
||||||||||||||||||||                                            |8:40%
||||||||||||||||||||||||||||||||                                |13:65%
```

```
Cantidad de trabajadores:5
Graficando avance diario:
|||||||||||||||||||||||||||||||||||||||                         |16:80%
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||  |19:95%
|||||||||||||||                                                 |6:30%
||||||||||||||||||||||||||||||                                  |12:60%
||||||||||||||||||||||||||||||||                                |13:65%
```

3. Para calcular el máximo común divisor de dos números enteros se puede aplicar el algoritmo de Euclides, que consiste en ir restando el más pequeño del más grande hasta que queden dos números iguales, que serán el máximo común divisor de los dos números.

   Por ejemplo, si comenzamos con el par de números 412 y 184, tendríamos:

412	228	44	44	44	44	44	36	28	20	12	8	**4**
184	184	184	140	96	52	8	8	8	8	8	4	**4**

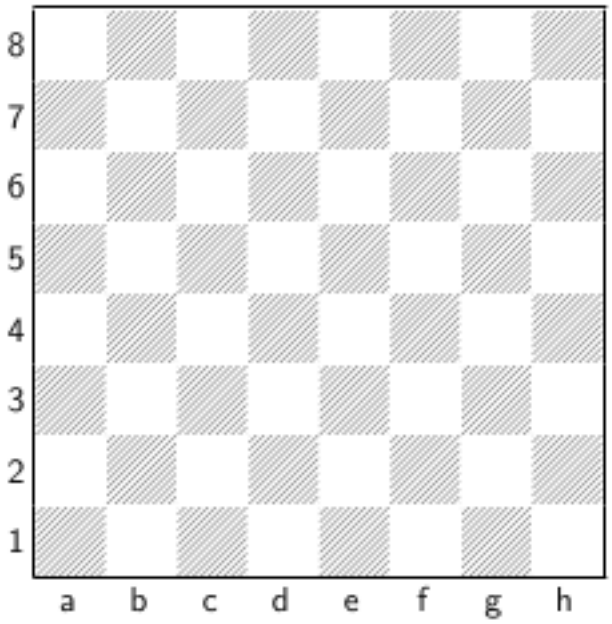
   Es decir, MCD(412, 184) = 4

   Se le pide diseñar una función recursiva para resolver este problema. Algunos ejemplos de diálogo de este programa serían:

```
Ingrese primer numero:18
Ingrese segundo numero:12
MCD:6
```

```
Ingrese primer numero:11
Ingrese segundo numero:10
MCD:1
```

1. Diseñar y Escribir un programa que permite detectar si la ubicacion de un ficha de ajedres se encuentra en alguna de las direcciones en que se mueve la reina, considerando que las columnas se representan por letras desde la 'A' hasta la 'H' de izquierda a derecha y que las filas por numeros de 1 al 8 de abajo hacia arriba como se muestra en la imagen. El usuario del programa debera ingresa 2 posiciones, la primera representa la ubicación de la reina y la segunda la ubicación de la otra pieza de ajedres, el programa debera devolve, "Puede ser tomada por la Reina" en caso se encuentre en alguna de las direcciones de la reina, o "No puede ser tomada por la reina" en caso contrario:



Algunos ejemplos del funcionamiento de este programa serían:

Listing 1: Ejemplo 1

```
Ingrese la ubicacion de la reina: A2
Ingrese la ubicacion de la otra pieza de ajedres: B5

No puede ser tomada por la Reina
```

Listing 2: Ejemplo 2

```
Ingrese la ubicacion de la reina: D3
Ingrese la ubicacion de la otra pieza de ajedres: f5

Puede ser tomada por la Reina
```

Listing 3: Ejemplo 3

```
Ingrese la ubicacion de la reina: A3
Ingrese la ubicacion de la otra pieza de ajedres: a3

Error las piezas de ajedres no pueden estar en la misma
   celda
```

Listing 4: Ejemplo 4

```
Ingrese la ubicacion de la reina: Z3
Error las piezas de ajedres no pueden estar en la fuera
   del tablero
Ingrese la ubicacion de la reina: A3
Ingrese la ubicacion de la otra pieza de ajedres: X2
Error las piezas de ajedres no pueden estar en la fuera
   del tablero
Ingrese la ubicacion de la otra pieza de ajedres: A2

Puede ser tomada por la Reina
```

2. Diseñar y escribir una función (esta en doble) que reciba un número (N) y que verifique si el valor doble (N*2) incluye alguno de los dígitos del valor original N, la función debe retornar el valor true o false, si devuelve true entoces imprimir "N*2 incluye digitos de N",sino"N*2 NO incluye digitos de N"donde N es el número ingresado:

   Algunos ejemplos de este programa serían:

Listing 5: Ejemplo 1

```
Ingrese un numero: 123321
246642 incluye digitos de 123321.
```

Listing 6: Ejemplo 2

```
Ingrese un numero: 732
1464 NO incluye digitos de 732.
```

3. Diseñar y escribir una función recursiva (contar vocales)_ que reciba un texto y que cuente cuantas vocales hay en la palabra:

   Algunos ejemplos de este programa serían:

Listing 7: Ejemplo 1

```
Ingrese un texto: utec
2 vocales
```

Listing 8: Ejemplo 2

```
Ingrese un texto: murcielago
5 vocales.
```

Listing 9: Ejemplo 3

```
Ingrese un texto: nnnn
0 vocales.
```

Listing 10: Ejemplo 4

```
Ingrese un texto: aaaa
4 vocales.
```