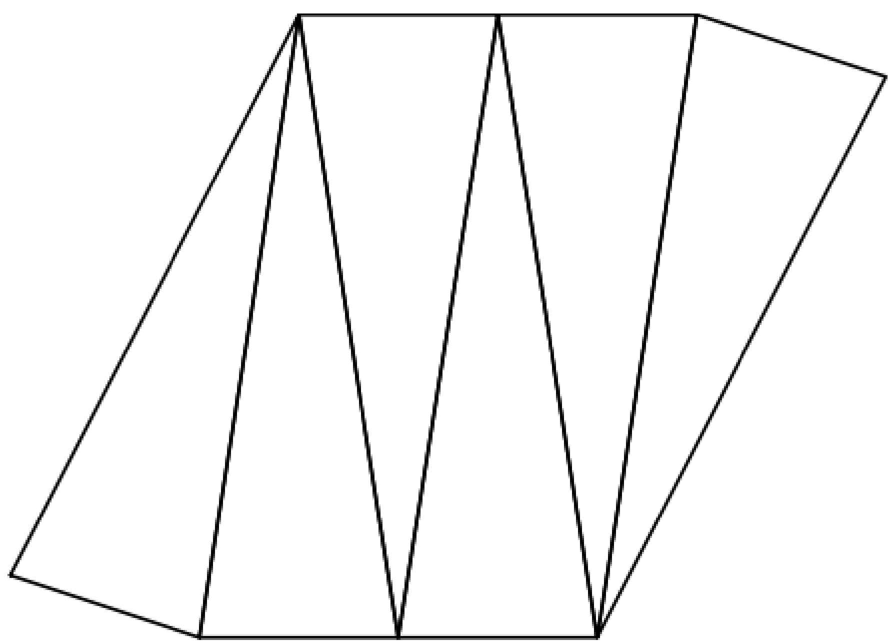
Solucionario

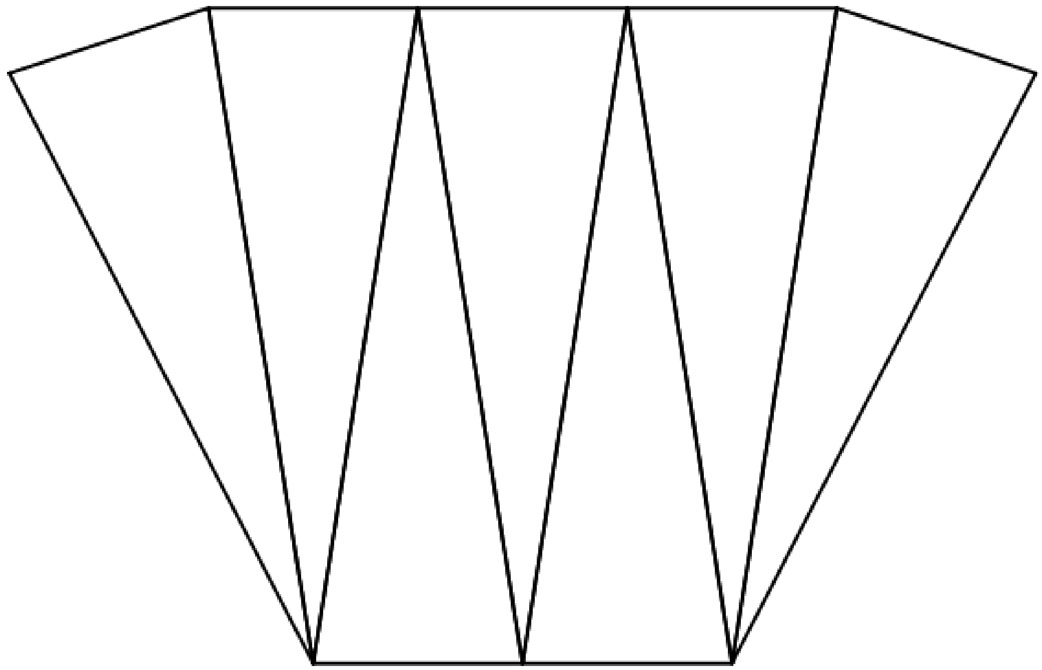
Matemática

1. Principio de Inclusión y exclusión: Sea T la cantidad total de números entre 1000 y 2023, y definamos a $F(i)$ como la cantidad de múltiplos de i en ese mismo rango. La respuesta será: $T - F(2) - F(3) - F(7) + F(6) + F(21) + F(14) - F(42) = 292$.

2. Para probar lo pedido podemos encontrar una construcción de un hexágono para todo $n \geq 6$Para n par podemos partir de la siguiente construcción con $n = 6$:

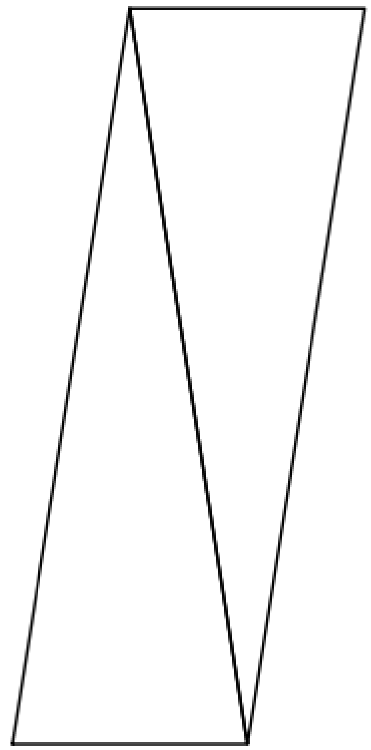


Para n impar podemos partir de la siguiente construcción con $n = 7$:



Las construcciones anteriores están formados por triángulos isósceles congruentes cuyo ángulo desigual sea menor a 30.

Ahora podemos poner la siguiente figura formada por 2 triángulos (también congurentes a los anteriores) en el medio de las anteriores construcciones para así formar una construcción para n par e impar con $6 \leq n$.



Por lo tanto para $n \geq 6$ podemos encontrar un hexágono convexo que puede ser divido en n triángulos congruentes.

3. Sea $A = \{a_1, a_2, a_3, a_4, a_5\}$ un conjunto de 5 números enteros. Definamos la secuencia P_i de la siguiente manera:

$$P_1 = a_1$$
$$P_2 = a_1 + a_2$$
$$P_3 = a_1 + a_2 + a_3$$
$$P_4 = a_1 + a_2 + a_3 + a_4$$
$$P_5 = a_1 + a_2 + a_3 + a_4 + a_5$$

De esta secuencia obtenemos dos observaciones:

a. P_i es la suma de elementos de un subconjunto de A.

b. Sea $1 \leq i < j \leq 5$, entonces podemos comprobar que

$$P_j - P_i = a_j + a_{j-1} + \cdots + a_{i+1},$$

este resultado también es la suma de elementos de un subconjunto de A.

Si algún P_i es múltiplo de 5, entonces ya encontramos el subconjunto requerido.

Supongamos que ningún P_i es múltiplo de 5. Entonces existirán P_i y P_j $(i < j)$ tal que ambos sean múltiplo de 5 + k para algún $k \in [1, 4]$ (esto sucede ya que existen 5 elementos P_i y solo 4 valores posibles de k, a este idea se le conoce como el *teorema del palomar*). Si restamos $P_j - P_i$ nos resultará un número múltiplo de 5, y por (b), también es la suma de elementos de un subconjunto de A.

Con la misma idea, podrías demostrar el caso general. Dado un conjunto de N números, siempre existe un subconjunto tal que la suma de sus elementos es múltiplo de N.

4. Tenemos que contar la cantidad de números menores a $2^{16} + 2^{15}$ que en binario se representa como: 11000000000000000, para esto tenemos 2 casos:
   - que el número sea mayor o igual a 2^{16}, para este caso los primeros 2 bits más significativos serían 10 luego tenemos que escoger 4 de los siguientes 15 bits siguientes para que sean 1, esto se puede hacer de $\binom{15}{4} = 1365$ formas.
   - que el número sea menor a 2^{16}, para este caso solo es escoger 5 bits de 16 bits (potencias de 2 con exponentes del 0 al 15) para que sean 1, esto se puede hacer de $\binom{16}{5} = 4368$ formas.

   Por lo tanto la respuesta es $1365 + 4368 = 5733$.

5. Primero demostraremos que entre 6 monedas consecutivas siempre habrá al menos una de 50 céntimos.

Prueba 1:

Supongamos que esta afirmación es falsa, ósea existen 6 monedas consecutivas donde ninguna es de 50 céntimos. Esta claro que las 6 monedas no pueden ser de 10 céntimos, por tanto debe haber al menos una de 20, ahora analizando por casos:

- Si hay exactamente una moneda de 20, al tomar un bloque de 5 monedas que contega la de 20 se tendrá una suma de 60 lo cual no puede ser.
- Si hay 2 monedas de 20 pero que están juntas, entonces no puede haber el patrón 10 20 20 10 por lo que el bloque de 6 monedas empieza con 20 20 10, a la derecha del número 10 debe haber un 20 20 para que no haya suma 60, ahora tenemos 20 20 10 20 20 pero el sexto número no puede ser ni 10 ni 20 lo cual no puede ser.
- Si hay 2 monedas de 20 que no estan juntas, vemos que entre estas 2 monedas solo puede haber 1 o 3 monedas de 10, para el primer caso tenemos 20 10 20 y vemos que al costado de las de 20 tiene que haber una de 20 por lo que se lleva al caso donde hay 2 de 20 juntas que por el analisis anterior sabemos que lleva una a una contradicción, en el segundo caso tenemos 20 10 10 10 20 donde vemos que el sexto número no puede ser ni 10 ni 20 lo cual no puede ser.

Por lo tanto entre 6 monedas consecutivas siempre habra una moneda de 50 céntimos ya que en caso contrario se llega a una contradicción.

Prueba 2:

Podemos usar el siguiente lema que es una generalización del problema 3 para n números.

$lema$*:* Dada una secuencia de n números enteros, existe un bloque de términos consecutivos cuya suma es múltiplo de n.

$prueba$*:* Podemos usar la idea del problema 3. Sea $a_1, a_2, a_3, ..., a_n$ una secuencia de n numeros enteros. Considerando las sumas de prefijos P_i:

$$P_0 = 0,\ P_1 = a_1,\ P_2 = a_1 + a_2, ..., P_n = a_1 + a_2 + a_3 + ... + a_n$$

Por el $\mathbf{Principio\ del\ Palomar/Casillas}$ ($n + 1$ elementos, n casillas) dos de estas sumas son congruentes en módulo n, luego la diferencia de estas sumas será multiplo de n, y sabemos que la diferencia de dos sumas de prefijos es una suma de algunos términos consecutivos de la secuencia.

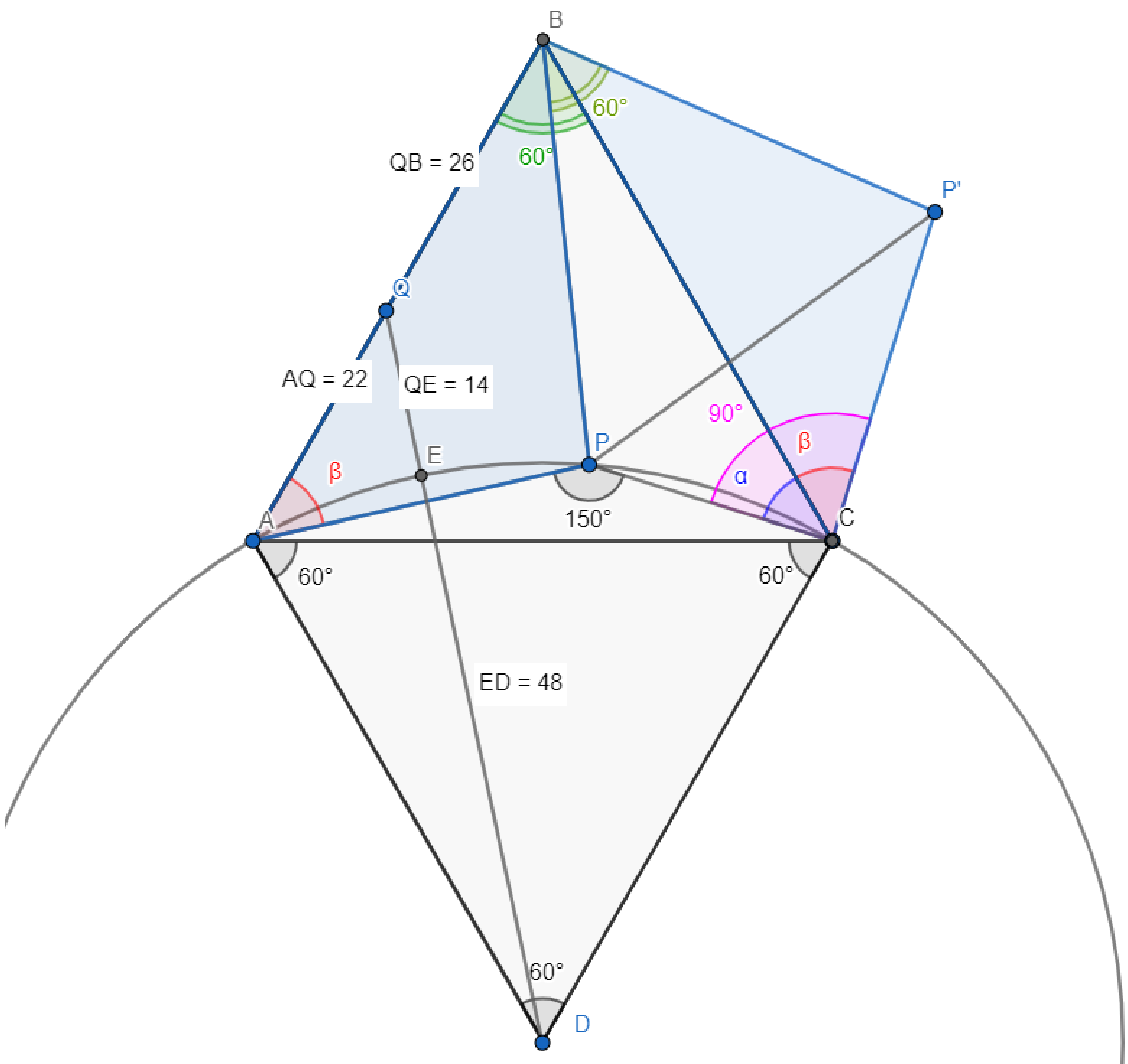
Ahora por el lema tomando 6 elementos, existe un bloque de términos consecutivos cuya suma sea multiplo de 6, si los valores son solo 10 o 20 entonces las posibles sumas multiplos de 6 son 60 y 120, y para el caso donde sea 120 sólo será cuando hay 6 monedas de 20 consecutivas que tambien contiene un bloque donde suma 60 agarrando 3 monedas de 20, por lo que siempre se tendra un bloque de monedas consecutivas que sumen 60 centimos si solo tomamos monedas de 10 y 20.

Finalmente como $101 = 16 \cdot 6 + 5$ podemos dividir las monedas en 16 grupos de 6 monedas y un grupo de 5 monedas, como sabemos que entre 6 monedas almenos 1 de 50 centimos, entre 101 monedas debe haber almenos 16 de 50 centimos. Para concretar el problema podemos mostrar un ejemplo con el siguiente patrón:

$$20, 10, 10, 10, 20, 50, 20, 10, 10, 10, 20, 50, ..., 50, 20, 10, 10, 10, 20$$

6. Veamos 2 posibles soluciones al problema:

Solucion geometrica



Dado la figura sea α y β la medida de los angulos PCB y PAB respectivamente, así también tenemos el dato que $PA^2 + PC^2 = PB^2$, para aprovechar este último dato podemos hacer que estos valores sean lados de algún triángulo. Si rotamos el triángulo ABP alrededor del punto B tal que el punto A este en C obtenemos el triangulo CBP', ahora como $BP = BP'$ y $\angle ABC = \angle PBC = 60^\circ$ entonces el triangulo BPP' es equilatero.

Podemos notar que como $PP' = PB$ y $AP = CP'$ entonces $PC^2 + CP'^2 = PP'^2$ por lo tanto como se cumple el teorema de Pitágoras tenemos que $\angle PCP' = 90^\circ$, ademas dado que $\angle BAP = \angle BCP' = \beta$ tenemos que $\angle PCB + \angle BCP' = \angle PCP' \Leftrightarrow \alpha + \beta = 90^\circ$ luego $\angle BAP + \angle ABC + \angle BCP = \angle APC \Leftrightarrow \angle APC = \alpha + \beta + 60^\circ = 150^\circ$

Ahora que sabemos que el ángulo $\angle APC$ es constante e igual a 150°, podemos trazar el lugar geométrico de P que es el arco AC de un círculo centrado en D que pasa por A y B, donde D es un punto tal que ADC es equilátero. La razón por la que este arco es el lugar geométrico de P es porque cualquier punto X perteneciente al arco cumple que $\angle AXC = 150^\circ$ y como probamos esto hace que $XA^2 + XB^2 = XB^2$.

Ahora notamos que de todos los P posibles el que minimiza PQ es el cual D, P y Q sean colineales, este punto sería donde esta ubicado E en la figura, para calcular la respuesta solo hallamos DQ por ley de cosenos en QAD y le restamos 48 asi:

$$DQ^2 = AQ^2 + AD^2 + AQ \cdot AD$$
$$DQ^2 = 22^2 + 48^2 + 22 \cdot 48$$
$$DQ = 65$$
$$EQ = DQ - ED$$
$$EQ = 62 - 48$$
$$EQ = 14$$

Por lo tanto nuestro mínimo valor entero de PQ es 14 y es cuando $P = E$.

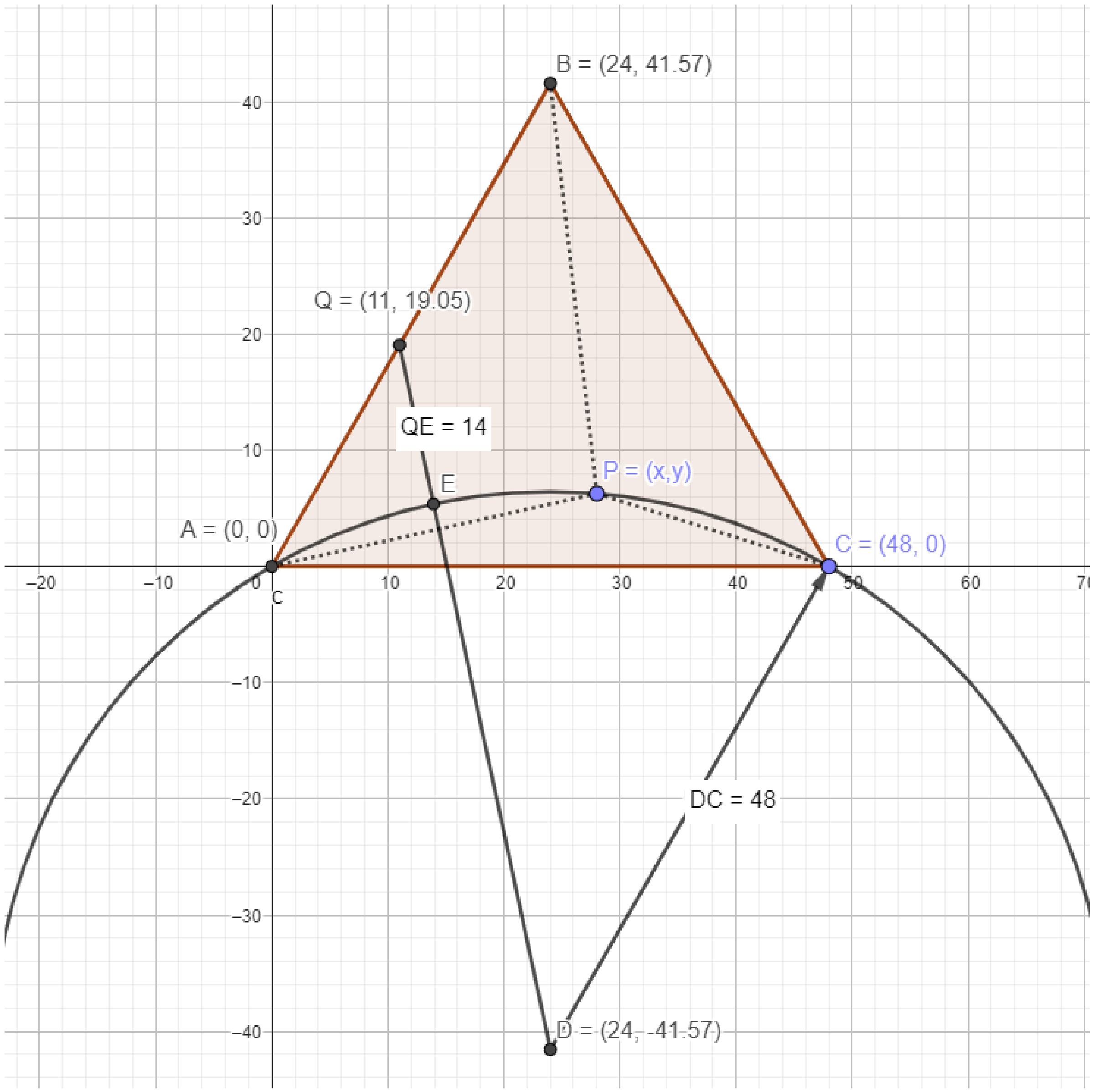
Solucion analitica

Si ubicamos el triangulo ABC en el plano cartesiano de tal forma que los vertices A, B y C sean $(0, 0)$, $(24, 24\sqrt{3})$, $(48, 0)$ respectivamente.

Ahora si definimos $P = (x, y)$ podemos hallar cuales son los puntos que cumplen la propiedad $PA^2 + PC^2 = PB^2$ asi:

$$PA^2 = x^2 + y^2,\ PB^2 = (x - 24)^2 + (y - 24\sqrt{3})^2,\ PC^2 = (x - 48)^2 + y^2$$
$$x^2 + y^2 + (x - 48)^2 + y^2 = (x - 24)^2 + (y - 24\sqrt{3})^2$$
$$x^2 + y^2 - 48x + 48\sqrt{3}y = 0$$
$$(x - 24)^2 + (y + 24\sqrt{3})^2 = 48^2$$

esta ultima ecuación corresponde a una circunferencia con centro en $D = (24, -24\sqrt{3})$ y radio 48, dado esto podemos decir que P esta en esta circunferencia mientras este dentro de ABC, veamos esto en la figura:



Ahora podemos hallar PQ minimo de la misma manera que en el metodo anterior hallando el punto E mostrado en la figura.

Por lo tanto PQ minimo es 14 y es cuando $P = E$.

7. Podemos notar que al hacer la operacion de $+2$ o -2 en modulo 4 nos da el mismo resultado esto porque $+2 \cong -2 \ (mod\ 4)$, dicho esto podemos analizar como cambia nuestro conjunto de numeros cuando hacemos una operacion.

   Inicialmente tenemos el numero $\underbrace{999\ldots999}_{\text{1997 9's}}$ que es congruente a 3 en modulo 4, y queremos llegar a un estado donde todos los numeros sean 9 es decir todos congruentes a 1 modulo 4, analizemos este estado final, sea x un numero por el cual se genera dos 9s esto quiere decir que:

$$x = a \cdot b \to a + 2 \cong\ 9\ (mod\ 4),\ b + 2 \cong\ 9\ (mod\ 4)$$
$$a \cong\ 3\ (mod\ 4),\ b \cong\ 3\ (mod\ 4)$$
$$x \cong\ a \cdot b \cong\ 3 \cdot 3 \cong\ 1\ (mod\ 4)$$
$$x \cong\ 1\ (mod\ 4)$$

Notamos que en el estado anterior al final todos los numeros tambien son congruentes a 1 modulo 4, por tanto podemos decir que cualquier estado que pueda llegar al estado deseado tiene que tener todos los numeros congruentes a 1 modulo 4, pero el estado inicial tiene un unico numero congruente a 3 modulo 4 por lo cual nunca se podra llegar al estado final dado por el problema.

8. Probaremos que Pepe no puede saltar por todas las piedras del pantano, para esto asumamos lo contrario que es que puede saltar por todas las piedras, para esto Pepe en algún momento pasara por la piedra 13, dado que 13 no es múltiplo de 2 y solo hay 14 piedras, no se puede llegar desde ninguna piedra hacia él, por lo que esta tiene que ser la piedra por la que se empieze. Luego de algunos movimientos Pepe deberia llegar a la piedra 14 y solo puede llegar al 14 si antes estaba en el 7, solo puede llegar al 7 si antes estaba en el 9, solo puede llegar al 9 si antes estaba en el 11 y solo puede llegar al 11 si antes estaba en el 13, entonces los primeros movimientos serían:

$$13 \to 11 \to 9 \to 7 \to 14$$

Análogamente, luego de algunos movimientos Pepe debe llegar a la piedra 5 y solo puede llegar al 5 si antes estaba en el 7, antes en el 9, antes en el 11 y antes en el 13, entonces los primeros movimientos serían:

$$13 \to 11 \to 9 \to 7 \to 5$$

Pero esto no puede ser porque antes mencionamos unos primeros movimientos distintos lo cual es una contradicción.

Por lo tanto Pepe no puede saltar por todas las piedras.

Ahora si encontramos un ejemplo donde salte por 13 piedras podriamos concluir que el máximo de piedras diferentes por el cual Pepe puede saltar es 13.

Un ejemplo sería:

$$13 \to 11 \to 9 \to 7 \to 5 \to 3 \to 1 \to 2 \to 4 \to 8 \to 6 \to 12 \to 10$$

9. Podemos modelar el problema como un grafo, donde necesitamos eliminar 4 aristas tal que el grafo se mantenga conexo.

   Observamos que las 12 aristas forman 4 triángulos. Si eliminamos 2 aristas de un mismo triángulo el grafo deja de ser conexo. Por lo tanto debemos eliminar 1 arista de cada triángulo ya que así el grafo se mantendria conexo mediante el vertice central 1.

   Por lo tanto podemos eliminar las 4 aristas de $3 \cdot 3 \cdot 3 \cdot 3 = 81$ formas.

10. Podemos decir que existe un edificio en Korhal con altura minima, sea x esta altura, ahora sean a, b, c, d las alturas de los edificios adyacentes, por condicion del problema $x = (a + b + c + d)/4$, y ademas por lo establecido $x \leq (a + b + c + d)/4$, la igualdad de esto solo se cumple cuando $x = a = b = c = d$, por lo que ahora a, b, c, d tambien son alturas minimas, si aplicamos la condicion del problema nuevamente en a, b, c, d siempre obtendremos nuevos edificios con alturas minimas y asi sucesivamente llegando a toda la cuadricula, dado esto podemos concluir que todos los edificios tienen la misma altura.

Algoritmia

1. Como cada a_i es menor o igual a n, podemos crear un *array* de frecuencias de los elementos (nos indicará cuántas veces aparece el elemento x en el arreglo dado).

```
#include<iostream>
using namespace std;

int main() {
    int n; cin >> n;

    int cnt[n+1];

    for(int i = 0; i <= n; i++) cnt[i] = 0;

    for(int i = 0; i < n; i++) {
        int a; cin >> a;
        cnt[a]++;
    }
```

```
    int ans = -1;

    for(int i = 0; i <= n; i++) {
        if(cnt[i] >= 3) {
            ans = i;
        }
    }

    cout << ans << "\n";

    return 0;
}
```

2. Cada rectángulo está representado por dos puntos. Sea un rectángulo $(A_x, A_y), (B_x, B_y)$ y otro $(C_x, C_y), (D_x, D_y)$; la intersección de ambos será $(\max(A_x, C_x), \max(A_y, C_y)), (\min(B_x, D_x), \min(B_y, D_y))$. Existen casos donde la intersección es nula, por lo que al final de obtener la intersección de los N rectángulos tenemos que verificar si el área es nula o no.

```
#include<iostream>
using namespace std;

int main() {
    int n; cin >> n;

    const int INF = (int) 1e9;
    int x1, y1, x2, y2;
    x1 = -INF;
    y1 = -INF;
    x2 = INF;
    y2 = INF;

    for(int i = 0; i < n; i++) {
        int xp1, yp1, xp2, yp2; cin >> xp1 >> yp1 >> xp2 >> yp2;

        x1 = max(x1, xp1);
        y1 = max(y1, yp1);
        x2 = min(x2, xp2);
        y2 = min(y2, yp2);
    }

    if(x1 > x2 or y1 > y2) cout << 0 << "\n"; // area nula
    else cout << (x2-x1) * (y2-y1) << "\n";

    return 0;
}
```

3. El problema busca la factorización de X que contenga el mayor número de factores, donde cada factor es al menos 2. Esta factorización debe ser, de hecho, el descomposición en factores primos de X, que probaremos por contradicción. Asumir que la lista óptima de enteros $Y_1, ..., Y_k$ contiene un número Y_i que es compuesto. En este caso, se puede factorizar aún más en Y_i = ab con a, b > 1. Reemplazando Y_i con los números a y b mantendrá invariante el producto de la lista (ya que Y_i = ab), pero amplía la longitud de la lista en 1, dando así una puntuación más alta secuencia. El único caso en que esto es imposible es si todos los Y_i son primos, entonces la lista que buscamos debe ser la descomposición en factores primos de X.

```
#include <iostream>
using namespace std;
int main(){
  int n; cin>>n;
  int cnt = 0;

  for(int i=2;i*i<=n;i++){
    while(n%i== 0 ){
      n/=i; //primos
      cnt++;
    }
  }

  if(n!=1)cnt++;

  cout<<cnt<<endl;
  return 0;
}
```

4. La idea principal del problema es analizar el caso en el cual dos hormigas se chocan. Según el problema, cuando dos hormigas se chocan, ambas cambian de dirección y continúan con su camino. Supongamos que todas las hormigas son exactamente iguales, y considera las dos siguientes situaciones, cuando dos hormigas se chocan:

   - ambas cambian de dirección,
   - se traspasan la una a la otra y cada una mantiene su dirección

   se puede observar que ambas situaciones generan el mismo resultado. Por lo que podemos considerar que cada hormiga traspasará a las demás e irá al extremo sin cambiar su dirección, por lo tanto, el tiempo en el que todas las hormigas abandonan la barra, será el tiempo de la hormiga que más se demora en salir de la barra.

```
#include <iostream>
using namespace std;
int main(){
  int n; cin >> n;
  int L; cin >> L;
  int a[n], d[n];

  for(int i = 0; i < n; i++) cin >> a[i];
  for(int i = 0; i < n; i++) cin >> d[i];

  int ans = 0;

  for(int i = 0; i < n; i++) {
    if(d[i] == 0) ans = max(ans, a[i]);
    else ans = max(ans, L-a[i]+1);
  }

  cout << ans << "\n";

  return 0;
}
```

5. Dada una cadena s de longitud n queremos encontrar una cadena t (con la misma longitud) tal que tenga la menor similitud posible con s. En el problema, nos dicen que la cadena s y t consisten de 4 caracteres: U, N, I y O. Simplifiquemos el problema y supongamos que las cadenas solo consisten en 2 caracteres: A y B. s_A representará la cantidad de caracteres A en s y s_B representará la cantidad de caracteres B en s. Análogamente, t_A y t_B representan la cantidad de caracteres en la cadena t. Sin pérdida de generalidad, consideremos que $s_A \leq s_B$.

   - Si t consiste de solo caracteres 'A', la similitud entre ambas cadenas será de s_A.
   - De igual manera, la similitud entre ambas será s_B, si t consiste de solo caracteres B.

   Como $s_A \leq s_B$, entonces la cadena que consiste solo de caracteres 'A' podría ser una posible respuesta y la que consiste de solo caracteres 'B' queda descartada. ¿Podemos conseguir una cadena t que tenga una similitud con s menor a s_A?

Ahora nuestro objetivo será demostrar si existe o no una cadena con similitud menor a -s_A. Analicemos los dos siguientes casos:

- Si $t_A \geq s_A$, entonces existirá una cadena común de solo caracteres A con longitud s_A, es decir la similitud será mayor o igual a s_A.
- Si $t_A < s_A$. Imaginemos que $t_A = s_A - k$. Recordemos que $t_A + t_B = n$ y $s_A + s_B = n$, operando y reemplazando en las ecuaciones, tendremos que $t_B = s_B + k$. Entonces existirá una cadena común de solo caracteres B de tamaño s_B, es decir la similitud será mayor o igual a s_B ($s_A \leq s_B$).

Podemos ver que en ambos casos, no existe una cadena con similitud menor a s_A, entonces podemos decir que la mínima similitud posible será s_A y este resultado se obtiene cuando la cadena t solo consiste de caracteres A, es decir el carácter que tenga menor frecuencia en la cadena s.

Con el mismo razonamiento, en el problema original (cuando las cadenas consisten de 4 posibles caracteres), se puede demostrar que la respuesta será cuando t consiste solo de aquel carácter que tenga menor frecuencia en s.

```
#include <iostream>
using namespace std;
int main(){
  int n; cin >> n;
  char s[n+1];
  int frec_u = 0, frec_n = 0, frec_i = 0, frec_o = 0;

  for(int i = 0; i < n; i++) {
    if(s[i] == 'U') frec_u++;
    if(s[i] == 'N') frec_n++;
    if(s[i] == 'I') frec_i++;
    if(s[i] == 'O') frec_o++;
  }

  int min_frec = n;

  min_frec = min(min_frec, frec_u);
  min_frec = min(min_frec, frec_n);
  min_frec = min(min_frec, frec_i);
  min_frec = min(min_frec, frec_o);

  char c;

  if(min_frec == frec_u) c = 'U';
  if(min_frec == frec_n) c = 'N';
  if(min_frec == frec_i) c = 'I';
  if(min_frec == frec_o) c = 'O';
```

```
  for(int i = 0; i < n; i++)
    cout << c;

  cout << "\n";
  return 0;
}
```