Criptografia bàsica (2)

Artur Travesa

(versió 2021-04)

Introducció general

L'origen d'aquestes notes es remunta a diferents cursos d'Aritmètica o de Criptografia, a càrrec de l'autor, per a estudiants de Matemàtiques o d'Informàtica de la Universitat de Barcelona.

La idea bàsica és descriure algunes aplicacions importants de l'Aritmètica bàsica (diguem, de nivell de primer curs) a les transmissions xifrades d'informació.

En cap cas es tracta d'un curs de Criptografia, que caldria encabir en espais més amplis de coneixements, que haurien d'incloure, probablement, parts de teoria de la comunicació, de complexitat algorítmica o computacional, d'aprenentatge automàtic, o d'estudi de teories de compartició de secrets, entre d'altres.

El format triat per a la presentació és el d'un *notebook* de *Mathematica*, per la facilitat que té aquest programari per a poder desenvolupar els càlculs no trivials de manera prou senzilla i entenedora, d'una banda, i per a permetre fer una presentació escrita prou raonable des del punt de vista de material escrit, de l'altra. En particular, la possibilitat d'incloure els càlculs dins del text de manera natural en fan una bona eina comunicativa i, alhora, facilita molt el càlcul amb exemples no trivials.

A fi de veure tot el contingut del *notebook* convé executar-lo. Això es pot fer de cop o bé, més recomanable, cel·la a cel·la a mesura que s'avança en la lectura i comprensió dels diferents continguts.

Observació: Per al cas en què no es disposi del programari, hi ha la versió executada del *notebook* en format pdf.

Amb la finalitat doble d'una banda, de no fer textos molt llargs o amb molts continguts, i de l'altra de poder ampliar de manera senzilla els continguts que s'hi tractin, el material s'ha dividit en diferents *notebooks*, que desrivim a continuació, en l'apartat de referències.

Referències

[Cripto- 1]: **Criptografia bàsica (1)**.
Travesa, A.: CriptografiaBasica-1; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Una iniciació a la codificació de missatges. El criptosistema de Cèsar. El criptosistema de Vigenère.

[Cripto- 2]: **Criptografia bàsica (2)**.
Travesa, A.: CriptografiaBasica-2; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Els criptosistemes lineals. Els criptosistemes afins. Sufixació de missatges. Farciment de missatges.

[Eratostenes]: **Un garbell d'Eratòstenes.**
Travesa, A.: Eratostenes; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Un garbell d'Eratòstenes.

[Cripto- 3]: **Primeritat. Construcció de primers**.
Travesa, A.: ConstruccioDePrimers; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Test de primeritat de Solovay-Strassen. Test de primeritat de Miller-Rabin. Un certificat congruencial de primeritat. Construcció certificada de nombres primers de mida prefixada. Aplicació al càlcul de claus RSA. Aplicació (exercici) al càlcul de claus ElGamal.

[Cripto- 4]: **Factorització**.
Travesa, A.: Factoritzacio; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Un garbell d'Eratòstenes. Tests de primeritat de Solovay-Strassen i de Miller-Rabin. Un certificat congruencial de primeritat. Un algoritme bàsic de divisó per nombres primers petits. Un algoritme bàsic de divisó per nombres petits. El mètode de factorització de Fermat. El mètode de factorització p-1 de Pollard. El mètode de factorització rho de Pollard.

[RSA]: **Criptosistemes de tipus RSA**.
Travesa, A.: RSA; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Una descripció bàsica dels criptosistemes de tipus RSA: les claus; xifratge; desxifratge; observacions sobre la seguretat.

[ElGamal]: **El criptosistema ElGamal**.
Travesa, A.: ElGamal; accessible en forma notebook o en format pdf des de https://travesa.cat/notes/
Contingut: Logaritmes discrets. Una descripció bàsica del criptosistema ElGamal: el grup cíclic; les claus; xifratge i desxifratge.

[Tr-1]: Travesa, A.: *Aritmetica*. Edicions de la Universitat de Barelona, col·lecció UB, n. 25. Barcelona, 1998. ISBN:84-8338-031-5.

6. Criptosistemes lineals o afins

- **Exercici 2.1**

El criptosistema de Cèsar és un criptosistema de translació. De fet, el conjunt de caràcters es posa en bijecció amb $\mathbb{Z}/N\mathbb{Z}$, on N és el nombre de caràcters possibles (en el nostre cas, hem usat N=65536). Una clau qualsevol del criptosistema correspon a un element c de $\mathbb{Z}/N\mathbb{Z}$ I els missatges plans (i els missatges xifrats) són els elements m de $\mathbb{Z}/N\mathbb{Z}$.
El procés de xifratge és efectuar la translació x(m):=m+c (mod N); i el procés de desxifratge és la translació x'(m'):=m'+(N-c) (mod N).

No cal limitar-nos a aplicacions tan senzilles com les translacions; podem utilitzar transformacions lineals x(m):=c*m (mod N), o bé afins x(m):=c*x+d (mod N), on ara, les claus són c, per a les lineals, o la parella (c,d), per a les afins.

Però cal que les aplicacions de xifratge i les de desxifratge siguin bijectives (de fet, cal que l'aplicació de xifratge seguida de la de desxigratge sigui la identitat; per tant, l'aplicació de desxifratge ha de ser exhaustiva i l'aplicació de xifratge ha de ser injectiva; i en un conjunt finit, això significa que les dues han de ser bijectives i l'una inversa de l'altra); per tant, cal que c sigui invertible mòdul N. En aquest cas, si c' és l'invers de c mòdul N, les transformacions inverses (o sigui, les de desxifratge), són les x'(m'):=c'*m' (mod N), i x'(m'):=c'(m'-d) (mod N), respectivament. Cal notar que, en particular, la transformació inversa d'una transformació lineal també és lineal; i la transformació inversa d'una transformació afí també és afí.

En efecte, x'(m')=c'(m'-d)=c'*m'+(-c'*d) (mod N).

(a) Quàntes claus possibles de xifratge hi ha per al criptosistema lineal amb N=65536?
(b) Quàntes claus possibles de xifratge hi ha per al criptosistema afí amb N=65536?
(c) Es demana escriure una funció **AfiX** per a xifrar amb el criptosistema afí de 65536 caràcters. (La funció ha d'admetre com a paràmetres el missatge i la clau de xifratge).
(d) Es demana escriure una funció **AfiDX** per a desxifrar amb el criptosistema afí de 65536 caràcters. (La funció ha d'admetre com a paràmetres el missatge i la clau de xifratge).
(e) Es demana comprovar que aquestes funcions xifren i desxifren correctament alguns missatges.

(a)

```
65 536 == 2^16
```

```
True
```

Resposta: N=65536=2^16; per tant, els elements invertibles mòdul N són els nombres senars entre 1 i N; i d'aquests n'hi ha la meitat; o sigui 2^15=32768.

(b)

La quantitat de parelles de nombres (c,d) tals que c és invertible mòdul N és el producte de 2^15 per 2^16; o sigui, 2^31=2147483648.

(c)

```
AfiX[m_String, {c_Integer, d_Integer}] := Module[{b = 2^16, lta = ToCharacterCode[m]},
  FromCharacterCode[Table[Mod[c lta[[i]] + d, b], {i, 1, Length[lta]}]]
 ]
```

(d)

```
AfiDX[m_String, {c_Integer, d_Integer}] :=
 Module[{b = 2^16, lta = ToCharacterCode[m], cc, dd},
  cc = PowerMod[c, -1, b];
  dd = Mod[-cc d, b];
  FromCharacterCode[Table[Mod[cc lta[[i]] + dd, b], {i, 1, Length[lta]}]]
 ]
```

(e)

```
pla = "Cal que les aplicacions de xifratge i de desxifratge siguin bijectives; per tant,
    cal que c sigui invertible mòdul N. En aquest cas, si c' és l'invers de c
    mòdul N, les transformacions inverses (o sigui, les de desxifratge), són les
    x'(m')=c' m' (mod N), i x'(m')=c'(m'-d) (mod N), respectivament. Noteu que,
    en particular, la transformació inversa d'una transformació lineal també és
    lineal; i la transformació inversa d'una transformació afí també és afí.";
```

```
xifrat = AfiX[pla, {3, 23456}]
```

```
■■■■■■■■■■■■■峰■■■■■■峭峪■■■■■■■■島■■■■■■■■■■■■■■■島■■■■■■■■■■峪■■■■■■■■■■■屑■:
   峰■島■■■峪■尤■■■■■■■■■■■■■■■■■■■峪■■島■■■■■■■并■■■■■■■屯峪■■■■■■■■■■■尤■■■■■■■■■■:
   ■■■峪■■島■■■■■■■■并■■■■■尤■■■■■■島■峪■■峭島■■■■峭峪■■■峪■■島■■■■■■峭■■■■■■尤■■■■■■■■■■:
   ■■■■■島■■■■■尤■■幹峪■■■■■■■■■■■■■■■■■■■■■峭■■■■尤■■■■■■■■■■■■■■■■■■■■■峭■■■■尤■:
   島■■峰■■■■■■■■峪■■■■峭■■■■■■■尤■■峪■峰■島■■■■■■島尤■■■■■島■峪■■峭島■■■■幹■■峪■■島■■■■■■■:
   峪■■■島■峪■■峭島■■■■幹■■■峪■■■■■■■■■■■■■■■■■峪■■■屑■■■■■■■■島■峪■■峭島■■■■幹■■峪■■島■■■■■■:
   ■峪■■■島■峪■■峭島■■■■幹■■■■■■■■■■■■■■■■■■
```

```
desxifrat = AfiDX[xifrat, {3, 23456}]
```

```
Cal que les aplicacions de xifratge i de desxifratge siguin bijectives; per tant, cal que c sigui invertible mòdul N. En
  aquest cas, si c' és l'invers de c mòdul N, les transformacions inverses (o sigui, les de desxifratge), són les
  x'(m')=c' m' (mod N), i x'(m')=c'(m'-d) (mod N), respectivament. Noteu que, en particular, la transformació inversa
  d'una transformació lineal també és lineal; i la transformació inversa d'una transformació afí també és afí.
```

```
desxifrat == pla
```

```
True
```

```
xifrat2 = AfiX[pla, {33 457, 23 456}]
```

```
■■■■■■■■■■■■■褐■■■■■■■■■■棄■■■■渦蹲■■■■■■■棄■■棄■■■■渦蹲■■■■■■■■■■■■■■■■■■■■■■穫■⋱
  褐■蹲■■■■■■■■■■■■■■■■■■■■■■■■■■■蹲■■■■■■ý■棄■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■⋱
  ■■■■■■蹲■■棄■■■■ý■棄■■■■■■■■■■■蹲■■■渦■蹲ý■■■■■■■■■■■蹲■■■■■■■■■■■■■■■■■■棄■■棄■⋱
  ■■■渦蹲■■■■■■■■■■■■■■■■■■ý■■■■■■ý■■■ý■棄■■■■■■■■■■■ý■■■■■■ý■咽棄■■■ý■棄■■■■■蹲■■⋱
  褐■■■■■■ý■■■■■■■■■■■■■■■■■■■褐■蹲■■■■■■蹲■■■■■■蹲■■■渦■蹲ý■■■■■■■■■蹲■■■棄■■■■■■⋱
  蹲■■■渦■蹲ý■■■■■■■■■■■■■■ý■■■■■■■■■■■■■穫■■■■■■■蹲■■■渦■蹲ý■■■■■■■■■蹲■■■棄■■■■■■⋱
  蹲■■■渦■蹲ý■■■■■■渦好■■■ý■■■■■■■渦好■
```

```
desxifrat2 = AfiDX[xifrat2, {33 457, 23 456}]
```

```
Cal que les aplicacions de xifratge i de desxifratge siguin bijectives; per tant, cal que c sigui invertible mòdul N. En
  aquest cas, si c' és l'invers de c mòdul N, les transformacions inverses (o sigui, les de desxifratge), són les
  x'(m')=c' m' (mod N), i x'(m')=c'(m'-d) (mod N), respectivament. Noteu que, en particular, la transformació inversa
  d'una transformació lineal també és lineal; i la transformació inversa d'una transformació afí també és afí.
```

```
desxifrat2 == pla
```

```
True
```

- **Exercici (extra)**

Es demana recuperar el criptosistema de Cèsar de **[Cripto- 1]** i comprovar que, per a una clau c, coincideix amb el cas particular d'un criptosistema afí de clau (1,c).

```
Cesar[missatge_String, clau_Integer] := Module[{b = 2^16, lta = ToCharacterCode[missatge]},
  FromCharacterCode[Table[Mod[lta[[i]] + clau, b], {i, 1, Length[lta]}]]]
```

5. Sufixació i farciment de missatges

- **Observació**

Per a aquest tema, és convenient haver llegit el tema de codificació. Recuperem aquí les funcions necessàries de **[Cripro- 1]**.

```
Codifica[m_String] := Module[{lta = ToCharacterCode[m]},
  Flatten[Table[IntegerDigits[lta[[i]], 2, 16], {i, 1, Length[lta]}]]]
```

```
Descodifica[lta_List] := Module[{x},
  x = Partition[Flatten[Prepend[lta, Table[0, {i, 1, Mod[-Length[lta], 16]}]]], 16];
  FromCharacterCode[Table[FromDigits[x[[i]], 2], {i, 1, Length[x]}]]]
```

- **Sufixos**

Els criptosistemes que hem tractat a la secció anterior són de xifratge senzill: el xifratge es produeix caràcter a caràcter. De fet, són criptosistemes que s'anomenen monoalfabètics de substitució (cada caràcter de l'alfabet se substitueix per un altre caràcter del mateix alfabet).

En els criptosistemes de **xifratge per blocs**, el xifratge no es produeix caràcter a caràcter, sinó que el missatge es trenca a trossos, anomenats unitats de missatge o blocs de missatge, i les diferents unitats es xifren successivament. Els criptosistemes més senzills d'aquest tipus són els criptosistemes de permutació de les posicions i els criptosistemes afins de dimensió més gran que 1 (també anomenats de Hill). Tots ells usen unitats de missatge de longitud fixa, que pot ésser diferent per a cada criptosistema concret.

Exemple: "Un mi" "ssatg" "e tre" "ncat " "en bl" "ocs d" "e lon" "gitud" " cinc"

Si volem trencar un missatge en blocs de longitud fixa, *d* (*d* és la dimensió del criptosistema), cal que el nombre de caràcters del missatge sigui múltiple de *d*. En el cas que no ho sigui, cal afegir caràcters al final del missatge; d'això en direm "afegir-li un sufix" (o "sufixar-lo"). A més a més, per tal de dificultar la criptoanàlisi dels missatges, convé afegir els caràcters de manera aleatòria, i no pas d'una manera fixa o predeterminada.

D'altra banda, un cop desxifrem el missatge, cal que coneguem exactament on s'acaba el missatge original, a fi de poder treure-li el sufix que haguem afegit.

Doncs, caldrà afegir al missatge original, abans de xifrar-lo, la seva longitud, i fer-ho d'alguna manera que permeti, un cop desxifrat, recuperar exactament el missatge original.

Exercici 2.2 (Incorporació de la longitud del missatge)

Sovint se sol mesurar la longitud d'un missatge (o d'un fitxer, o d'una clau criptogràfica, etcètera) pel seu nombre de bits.
(a) Si utilitzem la longitud de tres dels nostres caràcters codificats (o sigui, 48 bits) per a escriure la longitud del missatge, quina longitud màxima pot tenir un fitxer (o un missatge) a fi que puguem escriure exactament la seva longitud?
(b) Es tracta de construir una funció **Sufixa** que, aplicada a un missatge pla (o bé codificat), m, de longitud acceptable, proporcioni el missatge m codificat i sufixat de manera que la longitud del missatge final sigui múltiple de 31*16=496 bits, dels quals els 3*16=48 darrers siguin l'expressió binària de la longitud del missatge codificat, i els caràcters restants, afegits entre el final del missatge i abans dels 48 bits de la longitud, siguin caràcters aleatoris; d'aquests caràcters aleatoris, el màxim és de 480=496-16 bits. (Notem que el missatge codificat ja conté una quantitat de bits múltiple de 16.)
Observació: Cal notar que cada vegada que se sufixa un mateix missatge pla, el sufix que s'hauria d'obtenir és diferent (aleatori), excepte el final, que correspon a la longitud del missatge pla.
(c) Es demana construir una funció **Recupera**, inversa de l'anterior, que proporcioni, a partir d'un missatge sufixat, el missatge pla original.

(a)

El càlcul següent ens diu que aquesta longitud màxima és de 32 Terabyte menys un bit:

```
2^(3×16)/(2^3×2^10×2^10×2^10×2^10)
```

```
32
```

O bé, si mesurem cada Kilobyte, Megabyte, Gigabyte o Terabyte com 1000 unitats de la mesura anterior, en lloc de les 1024=2^10, podríem representar longituds de l'ordre de 35,1844 Terabyte:

```
N[2^(3×16)/(2^3×10^3×10^3×10^3×10^3)]
```

```
35.1844
```

(b)

```
Sufixa[m_] := Module[{lta, x, long, y},
  If[Head[m] == String, lta = ToCharacterCode[m];
   x = Flatten[Table[IntegerDigits[lta[[i]], 2, 16], {i, 1, Length[lta]}]], x = m];
  long = Length[x];
  y = Table[RandomInteger[{0, 1}], {i, 1, Mod[-long - 48, 496]}];
  Flatten[Append[Append[x, y], IntegerDigits[long, 2, 48]]]
 ]
```

```
m = "Com és el sufix?"
```

```
Com és el sufix?
```

```
x = Sufixa[m]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0,
 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1,
 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1,
 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0,
 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1,
 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

```
Descodifica[x]
```

```
Com és el sufix?■■■恒■■■■■隔■■     ■j
```

```
Descodifica[Sufixa[m]]
```

```
Com és el sufix?■■■■■■■■■■■■     ■j
```

```
Descodifica[Sufixa[m]]
```

```
Com és el sufix?■■■■蠑■■豁■■■■     ■
```

```
Descodifica[Sufixa[m]]
```

```
Com és el sufix?■■■■僚■■■■■瑁     ■
```

```
Descodifica[Sufixa[m]]
```

```
Com és el sufix?■■■■■■■■■■■     ■
```

Notem que aquesta funcó cada vegada sufixa diferent; com cal, a causa del sufix aleatori.

(c)

```
Recupera[m_] := Module[{},
  Descodifica[Take[m, FromDigits[Take[m, -48], 2]]]
 ]
```

```
Recupera[Sufixa[m]]
```

```
Com és el sufix?
```

```
Recupera[Sufixa[m]] == m
```

```
True
```

■ **Exercici 2.3 (Farciment de missatges)**

Un dels resultats importants de criptografia implica que, a fi que hi hagi un bon grau de seguretat criptogràfica és condició necessària (encara que no suficient) que els missatges que es xifren siguin aleatoris.
Però els missatges plans que convé xifrar no sempre ho són, d'aleatoris. A fi d'apropar-los a aquesta propietat, convé farcir-los amb caràcters aleatoris.
(a) Es demana escriure una funció **Farceix** que, aplicada a un missatge pla (codificat i sufixat), m, proporcioni a la sortida un missatge de la manera següent:
primerament, ha de trencar el missatge m en unitats de missatge de 31 caràcters codificats (però no en unitats de 31*16=496 bits); després, per a cadascun d'aquests blocs de 31 caràcters, ha de construir un bloc de 63 caràcters codificats (pero no de 63*16=1008 bits) de manera que intercali cadascun dels 31 caràcters entre dos caràcters aleatoris. D'aquesta manera, cada bloc del missatge farcit començarà i acabarà amb un caràcter aleatori i, entremig, s'alternaran els caràcters del missatge i caràcters aleatoris.
(b) Es demana escriure una funció, **Buida**, inversa de la funció anterior; és a dir, que aplicada a una successió de blocs de 63 caràcters recuperi el missatge codificat i sufixat del qual prové el missatge farcit.
(c) Es demana aplicar aquesta funció al missatge següent (que és un missatge sufixat) i recuperar el missatge pla del qual prové.

```
missatge = {{{1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1}, {1, 1, 1, 0, 0, 0, 0, 1,
     0, 0, 1, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1},
    {1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
     1, 0, 1, 0, 1}, {1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0,
     0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {0, 0, 1, 1, 0, 0, 1, 1, 1,
     0, 1, 1, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0},
    {0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 1, 1, 0, 1, 0,
 0, 0, 1, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
{1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {0, 1, 1, 0, 0, 0, 1, 1, 0,
 0, 1, 0, 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0},
{0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1}, {0, 1, 0, 1, 1, 1, 0, 1, 1,
 0, 0, 0, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
{1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 1}, {0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {1, 0, 1, 0, 1, 1, 0, 1, 1,
 0, 1, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 1, 0, 0}, {1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1}, {1, 1, 0, 0, 1, 0, 1, 1, 1,
 0, 1, 0, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0}, {1, 0, 1, 1, 1, 0, 0, 0, 0,
 0, 1, 1, 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0}, {0, 0, 0, 1, 1, 1, 0, 1, 0,
 1, 1, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0},
{0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 1, 1}, {0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0}, {1, 1, 0, 0, 1, 0, 1, 0, 1,
 1, 0, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
  0, 1, 0, 0, 0, 0, 0}, {1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1}},
 {{1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 0, 0, 1}, {0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {1, 1, 0, 1, 0, 0, 1, 0, 1,
  0, 1, 1, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
 {0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0, 0}, {0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0}, {0, 0, 0, 1, 0, 1, 1, 1, 0,
  1, 1, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
 {1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 1, 0, 1}, {1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {1, 0, 1, 0, 0, 0, 0, 1, 0,
  1, 1, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
 {0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 0, 0, 1}, {1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {1, 1, 0, 1, 1, 1, 0, 0, 1,
  0, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0},
 {0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 1, 0, 1}, {0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {1, 0, 0, 1, 1, 0, 0, 0, 1,
  1, 0, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
 {0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 1, 0, 1}, {1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {0, 1, 1, 1, 1, 1, 1, 1, 1,
  1, 0, 1, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0},
 {0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 1, 0, 0, 1}, {1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {1, 1, 0, 1, 0, 1, 0, 1, 1,
  0, 1, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
 {1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 0, 0, 0}, {1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {0, 0, 0, 1, 1, 0, 0, 1, 0,
```

```
 0, 0, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0},
 {0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 0, 0, 1}, {0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {1, 1, 1, 1, 0, 1, 0, 0, 1,
  1, 0, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0},
 {0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0, 0}, {0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1}, {1, 0, 0, 1, 1, 0, 1, 0, 0,
  0, 1, 0, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1},
 {1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 1, 1, 0, 1}, {1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0}},
{{0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {0, 0, 1, 1, 1, 1, 0, 0, 1,
  1, 0, 0, 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1},
 {1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 1, 0, 1, 1, 0}, {1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1}, {1, 0, 0, 1, 0, 1, 0, 0, 0,
  0, 1, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
 {1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1}, {1, 1, 1, 1, 0, 1, 1, 0, 0,
  0, 1, 0, 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1},
 {1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 1, 0, 0, 0, 0, 0}, {1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1}, {1, 0, 0, 0, 1, 0, 1, 0, 0,
  1, 0, 0, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1},
 {1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
  1, 1, 0, 0, 1, 0, 1}, {1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1},
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {1, 0, 1, 0, 1, 1, 1, 1, 0,
  0, 0, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0},
 {1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
 1, 1, 0, 0, 0, 0, 1}, {1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 0, 1, 1, 0, 0, 0, 1, 0,
 1, 1, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0},
{1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 1, 1}, {1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {1, 0, 0, 1, 0, 1, 1, 0, 1,
 0, 1, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0},
{0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 0}, {0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1}, {1, 1, 1, 0, 1, 1, 0, 0, 0,
 0, 1, 0, 1, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
{1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 1, 0}, {1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {0, 0, 0, 0, 1, 0, 0, 0, 1,
 1, 1, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0},
{1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 1, 1, 1, 0, 0, 1, 0,
 0, 0, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1},
{0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0}},
{{1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 0}, {1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {0, 1, 0, 1, 1, 1, 0, 0, 1,
 0, 0, 1, 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
{0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 0, 1, 0}, {1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {0, 1, 0, 0, 1, 1, 1, 1, 0,
 0, 1, 1, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
{1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 0}, {1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1}, {0, 1, 0, 0, 0, 1, 0, 1, 1,
```

```
 1, 0, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0},
{0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {1, 0, 1, 1, 1, 1, 1, 0, 1,
 0, 1, 0, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0},
{1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 1}, {0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 0, 1, 0, 0, 0, 1, 1, 1,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
{1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 1, 0, 1}, {0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0}, {0, 1, 0, 0, 1, 0, 1, 1, 1,
 1, 0, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
{0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 1, 0, 0, 0}, {1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1}, {1, 1, 0, 0, 1, 0, 1, 1, 0,
 1, 1, 1, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
{1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0}, {1, 1, 1, 1, 1, 1, 0, 0, 1,
 0, 0, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
{1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0}, {0, 0, 1, 0, 1, 1, 0, 0, 0,
 1, 1, 0, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 1}, {1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 1, 1, 1, 0, 0, 0, 1, 0,
 0, 0, 1, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0},
{1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 1}, {0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0}},
{{0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
 0, 1, 0, 0, 0, 0, 0}, {1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {1, 1, 1, 0, 0, 0, 1, 0, 1,
 0, 1, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 0, 1, 1}, {0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {0, 0, 1, 1, 1, 1, 1, 1, 1,
 1, 0, 1, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0},
{1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 1}, {0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0}, {1, 0, 1, 1, 0, 0, 1, 0, 1,
 0, 1, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 0}, {1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {1, 0, 0, 0, 0, 1, 0, 0, 0,
 0, 0, 1, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1},
{1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0}, {0, 1, 1, 0, 0, 1, 1, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 1}, {0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1}, {1, 1, 1, 0, 0, 0, 0, 1, 1,
 0, 0, 0, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1},
{1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 0}, {0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1}, {1, 0, 1, 0, 1, 0, 0, 1, 0,
 0, 0, 0, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0},
{0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 0, 0, 1}, {0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1}, {0, 0, 0, 1, 1, 1, 1, 0, 0,
 0, 0, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
 1, 1, 1, 0, 0, 1, 0}, {0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 0, 1, 1, 1, 1, 1, 1, 0,
 1, 1, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 0, 0}, {1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {1, 1, 1, 0, 0, 1, 1, 0, 1,
 1, 1, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
{1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 0, 1}, {0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1}},
{{1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 0, 0, 1}, {1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {1, 1, 1, 0, 0, 1, 0, 1, 0,
 1, 1, 0, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1},
{0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 0, 0, 1}, {0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0}, {1, 0, 0, 1, 1, 1, 0, 0, 1,
 1, 1, 1, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1},
{1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 0, 1}, {0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1}, {0, 0, 0, 0, 1, 1, 1, 0, 0,
 0, 1, 1, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1},
{0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0}, {1, 1, 0, 1, 1, 1, 0, 1, 0,
 1, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 1, 0, 1,
 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0},
{0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 0, 0}, {0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0}, {1, 0, 0, 1, 0, 0, 0, 1, 1,
```

```
 1, 1, 1, 1, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
{0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 0, 1}, {1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1}, {1, 1, 0, 1, 0, 1, 0, 1, 1,
 1, 0, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1},
{0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0}, {1, 0, 1, 0, 0, 1, 0, 1, 1,
 0, 1, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1},
{0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 0, 1}, {1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {0, 0, 0, 1, 0, 0, 1, 1, 1,
 0, 1, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0},
{1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 0, 1, 0}, {1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {0, 0, 0, 1, 0, 1, 0, 0, 0,
 0, 1, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0},
{1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 0, 0, 1}, {1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1}},
{{0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 1, 1, 0}, {0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0}, {1, 0, 1, 1, 0, 0, 0, 0, 0,
 0, 1, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1},
{0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 1, 0, 1, 0, 1}, {0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1}, {0, 0, 1, 0, 0, 1, 0, 1, 1,
 0, 0, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
{0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
 1, 1, 0, 0, 1, 1, 0}, {0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1},
{0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1}, {1, 0, 0, 1, 0, 0, 0, 0, 0,
 1, 1, 0, 1, 0, 1, 0}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1},
{1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
     1, 1, 0, 1, 0, 0, 1}, {0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0}, {1, 1, 0, 1, 0, 1, 1, 1, 0,
     0, 1, 1, 1, 1, 0, 1}, {1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0},
    {1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0}, {0, 0, 1, 1, 1, 1, 0, 1, 1,
     1, 1, 0, 1, 1, 1, 1}, {1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0},
    {1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0}, {0, 1, 1, 0, 1, 0, 0, 0, 0,
     1, 0, 1, 0, 1, 1, 0}, {1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0},
    {1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1}, {1, 0, 0, 0, 0, 0, 0, 1, 0,
     1, 0, 0, 0, 0, 0, 0}, {1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1},
    {1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1}, {0, 0, 0, 1, 1, 1, 1, 0, 0,
     0, 0, 1, 1, 0, 1, 0}, {1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0},
    {1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1}, {0, 0, 0, 0, 1, 1, 0, 1, 1,
     1, 1, 0, 1, 0, 1, 1}, {1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1},
    {0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0}, {0, 0, 1, 0, 1, 1, 1, 1, 0,
     1, 0, 1, 0, 0, 0, 1}, {1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0}, {1, 0, 1, 1, 1, 1, 0, 0, 1,
     1, 0, 1, 0, 0, 0, 0}, {0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1},
    {1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0}, {0, 1, 0, 0, 1, 0, 0, 0, 0,
     1, 1, 0, 1, 1, 0, 1}, {1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1},
    {1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0}, {0, 0, 1, 1, 1, 0, 1, 0, 1,
     0, 0, 0, 0, 0, 0, 0}, {1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0},
    {0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0}, {0, 0, 1, 1, 1, 1, 1, 0, 1,
     0, 0, 0, 1, 0, 1, 1}, {1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0},
    {1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1}, {0, 1, 0, 0, 0, 1, 1, 1, 0,
     0, 1, 1, 1, 0, 1, 1}, {0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0},
    {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, {0, 1, 0, 1, 0, 1, 0, 0, 1,
     0, 1, 0, 1, 1, 0, 1}, {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
    {1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1}, {0, 0, 0, 0, 1, 1, 0, 0, 0,
     1, 0, 1, 0, 0, 0, 0}, {1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0}}};
```

(a)

```
Farceix[m_] := Module[{b = 2^16, long, lta, x},
  lta = Partition[m, 496];
  long = Length[lta];
  x = Table[Partition[lta[[i]], 16], {i, 1, long}];
  Table[Table[If[OddQ[j], IntegerDigits[RandomInteger[{0, b - 1}], 2, 16],
     x[[i]][[j / 2]]], {j, 1, 63}], {i, 1, long}]
 ]
```

```
s1 = Sufixa["Un missatge curt"];
```

```
f1 = Farceix[s1];
```

```
Length[f1]
```

```
1
```

```
s2 = Sufixa[
   "Un missatge molt més llarg, a fi que hi hagi molts bits en el missatge sufixat
     corresponent i, per tant, més d'un bloc en el missatge farcit final."];
```

```
f2 = Farceix[s2];
```

```
Length[f2]
```

```
5
```

(b)

```
Buida[m_] := Module[{},
  Flatten[Table[Table[m[[i]][[2 j]], {j, 1, 31}], {i, 1, Length[m]}]]
 ]
```

```
Recupera[Buida[f1]]
```

```
Un missatge curt
```

```
Recupera[Buida[f2]]
```

```
Un missatge molt més llarg, a fi que hi hagi molts bits en el
  missatge sufixat corresponent i, per tant, més d'un bloc en el missatge farcit final.
```

(c)

```
Recupera[Buida[missatge]]
```

```
Aquest és un missatge de prova que hauria de servir perquè comprovéssiu que la vostra funció de
  buidar els sufixos (i les de recuperar i descodificar els missatges) fa allò que hom pretén que faci.
```

```

```