Designing DNA barcodes

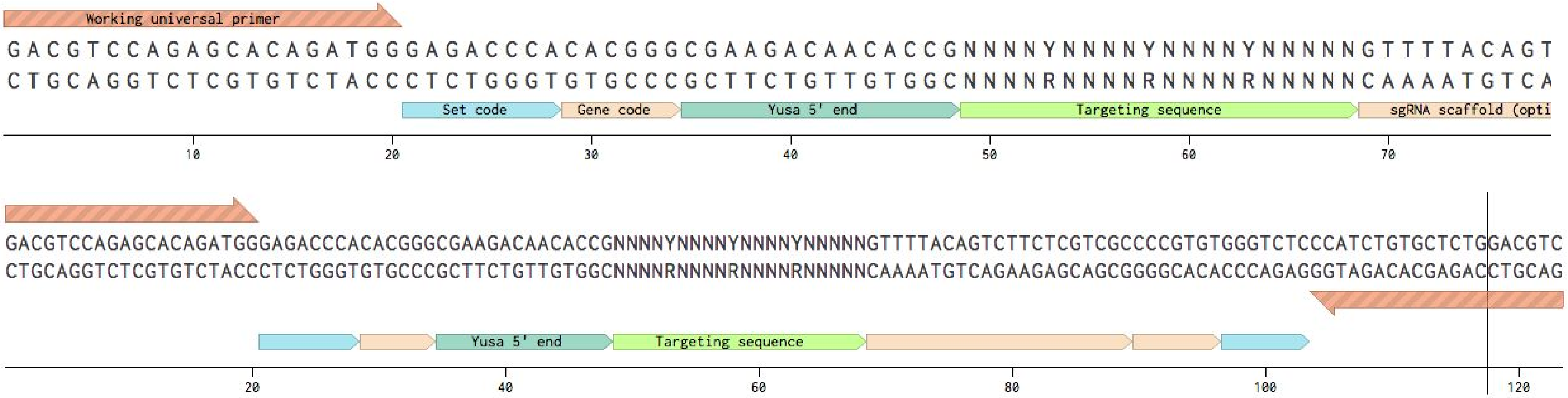
Izaak van Dongen - Work Experience

The problem

We can only order pools of thousands of oligos

It makes more sense to order a small set as a subset of a large pool

Members of a set need to be identifiable



The project - generating distinct barcodes

Generate a fixed number of barcodes with maximal distance, of a certain length

Generate a maximal number of barcodes with a fixed distance, of a certain length

Oligotm needs to be taken into consideration

Nested codes can be used to give more information

Generating codes

Simple ideas

Random codes with sieve

Hamming code

Hadamard code

Other ideas

Generating codes

Code at first written in Python for the command line

Simple ideas

Using all codes

- No tolerance for errors

Using random codes

- Not actually too bad..

The sieve

It greedily prints codes given to it, that aren't close to previously printed codes

It is less efficient and doesn't optimally select codes, and can only decode by lookup

It can more easily be applied to different instances of the problem (fixed number, maximise distance)

It is easier to adapt to constraints such as oligotm

No worries about binary to quaternary

Sample output

```
tctgca
ctcact
tgcacc
cccctc
agcaag
ctgaga
agctag
atgatg
caagtg
cgaaca -> python sieve.py 3 ->
taagga
caacac
ccggta
gcccct
aggtct
cgggga
cttgga
agcatt
tattct
gtaata
```

```
tctgca
ctcact
cccctc
agcaag
caagtg
aggtct
cgggga
gtaata
```

Hamming codes

They're a "designer code", less easy to vary

Easily correct errors

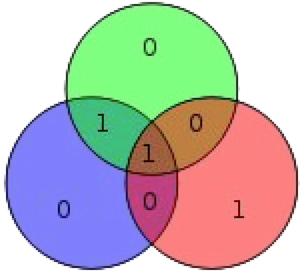
Generates lots of codes efficiently and reliable

They work for most required lengths

Hamming codes

Parity bits at each position which is a power of 2

A parity bit at position N covers data bits at positions which have a 1 where the parity bit has a 1 (ie N & databit_position is nonzero)



1001 -> 0011001

Parity bit #1 covers bits #3 (1), #5 (0) and #7 (1) so it is 1+0+1 (mod 2) -> 0

Parity bit #2 covers bits #3 (1), #6 (0) and #7 (1) so it is 1+0+1 (mod 2) -> 0

Parity bit #4 covers bits #5 (0), #6 (0) and #7 (1) so it is 0+0+1 (mod 2) -> 1

Error correction

0011001 -> 0011101

Parity bit #1 covers bits #3 (1), #5 (1) and #7 (1) so it should be 1+1+1 (mod 2) -> 1

Parity bit #4 covers bits #5 (1), #6 (0) and #7 (1) so it should be 1+0+1 (mod 2)

Parity bits #1 and #4 were incorrect, so the bit at position 1+4 = 5 must have changed

If a parity bit flips, the only parity bit which has an incorrect total will be that bit, so the sum of positions of incorrect bits will point to that bit

Hamming codes in base 4

Converting binary to base 4

00 -> A

01 -> C

10 -> G

11 -> T

If A becomes T then 00 becomes 11

2 errors can’t be detected

Hamming codes in base 4

Parity quads at each position which is a power of 2, showing sum modulo 4

0312 -> 1302312

The same procedure can be used to find the location of the error, and then as the sum has been stored, the difference between the stored sum and the received sum can be used to find the original value.

Application

Encode all of the possible data strings, from 0 to $4^n - 1$ (eg ‘000’ to ‘333’)

Translate this to DNA

One error can be detected and easily corrected

Sample output - encoding 64 strings of length 3

```
000000 010101 020202 030303 100110 110211 120312 130013
200220 210321 220022 230123 300330 310031 320132 330233
111000 121101 131202 101303 211110 221211 231312 201013
311220 321321 331022 301123 011330 021031 031132 001233
222000 232101 202202 212303 322110 332211 302312 312013
022220 032321 002022 012123 122330 132031 102132 112233
333000 303101 313202 323303 033110 003211 013312 023013
133220 103321 113022 123123 233330 203031 213132 223233
```

Can be directly converted to a string of bases

Hadamard codes

Also a designer code

Generate codes with large distance

Generates codes efficiently and reliable

They work for quite specific lengths

Hadamard codes

Rows of a hadamard matrix

Using Sylvester's construction

Each row of H and of -H is a codeword

$$H_1 = [1],$$

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

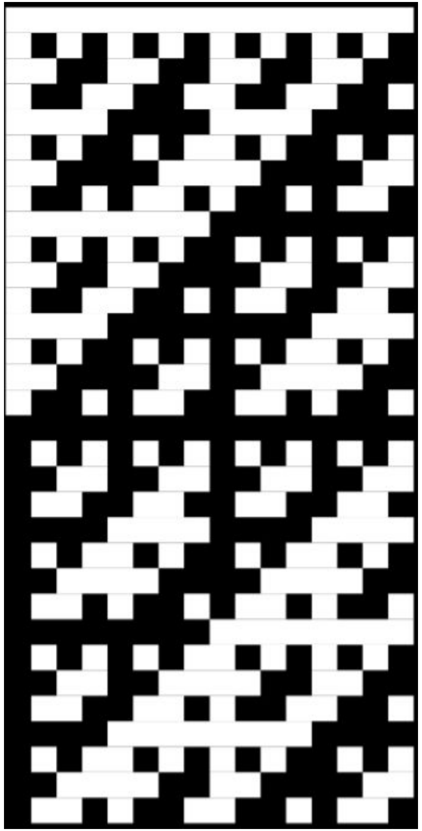
and

$$H_{2^k} = \begin{bmatrix} H_{2^{k-1}} & H_{2^{k-1}} \\ H_{2^{k-1}} & -H_{2^{k-1}} \end{bmatrix}$$

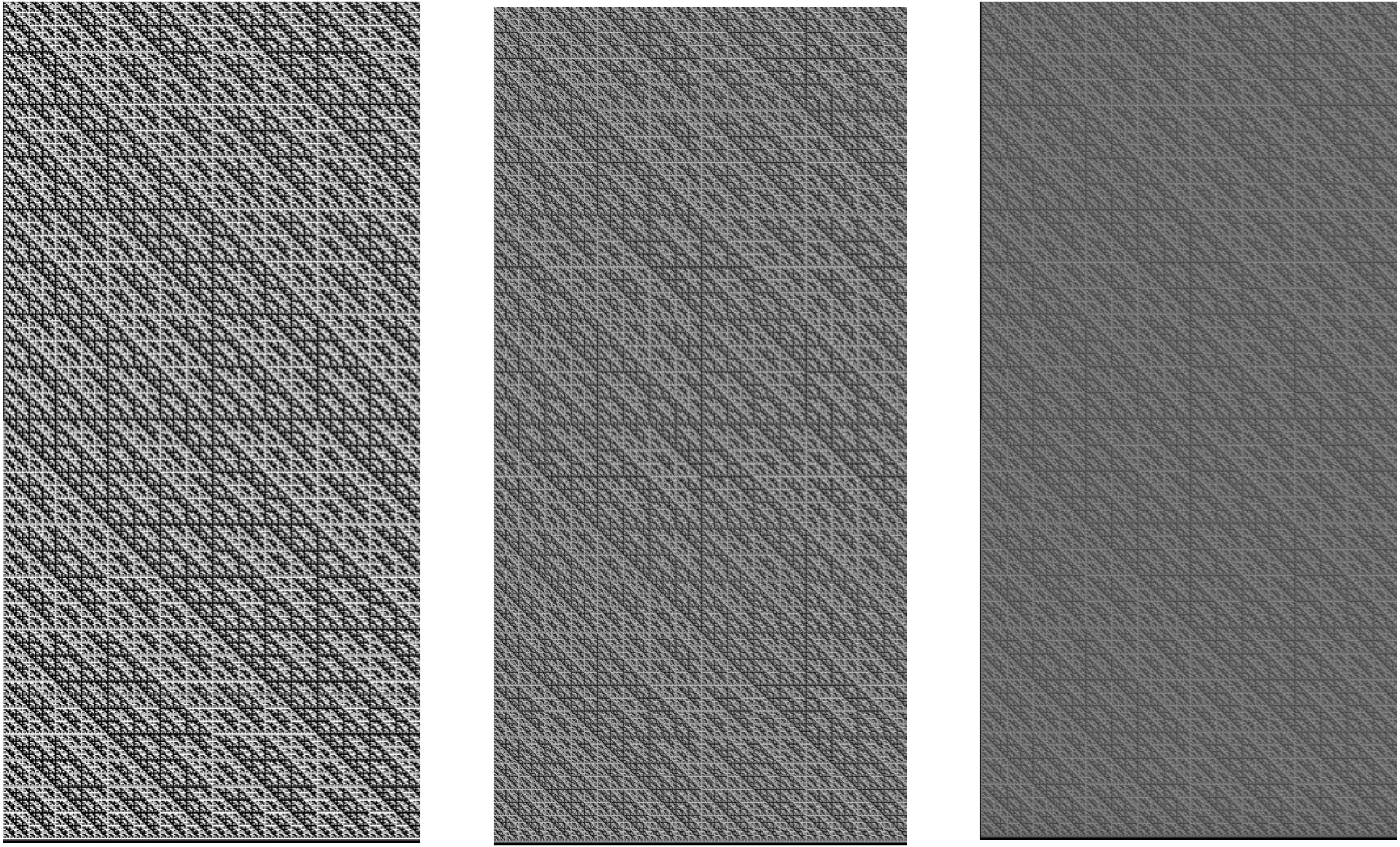
$$H4 = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{vmatrix}$$

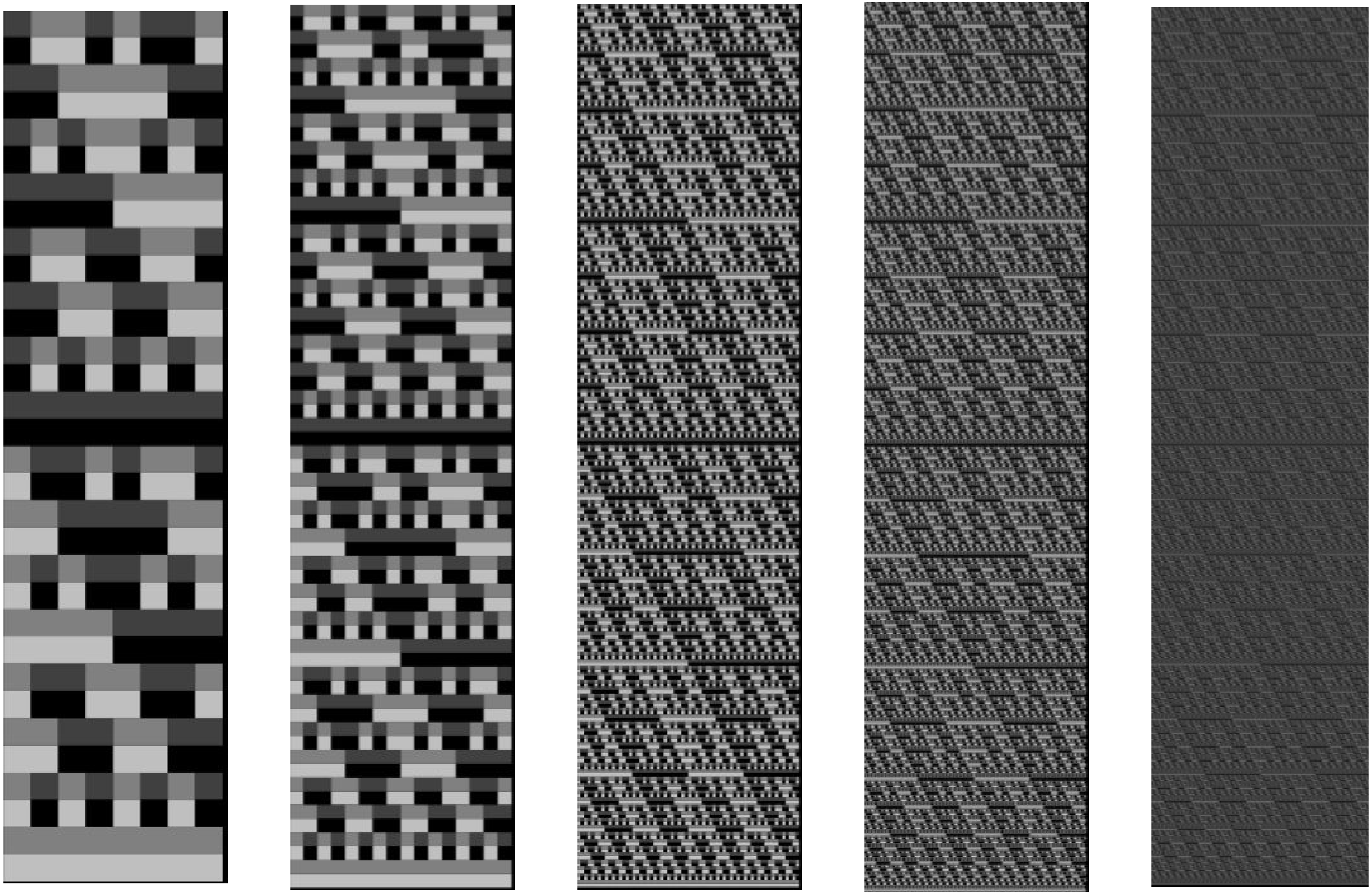
Sample output - H16, -1 becomes 0

1111111111111111	0000000000000000
1010101010101010	0101010101010101
1100110011001100	0011001100110011
1001100110011001	0110011001100110
1111000011110000	0000111100001111
1010010110100101	0101101001011010
1100001111000011	0011110000111100
1001011010010110	0110100101101001
1111111100000000	0000000011111111
1010101001010101	0101010110101010
1100110000110011	0011001111001100
1001100101100110	0110011010011001
1111000000001111	0000111111110000
1010010101011010	0101101010100101
1100001100111100	0011110011000011
1001011001101001	0110100110010110



Can be converted to a string of bases by each pair of binary digits, as it corrects so many errors





Other ideas

Using backtracking (similar to 8 rooks)

BK-tree sieve

Generating strings sequentially further apart, randomly or not

Golay codes

Polynomial codes

Web interface

Generating Hamming codes

Decoding Hamming codes

Generating Hadamard code

Decoding Hadamard codes

Sieving

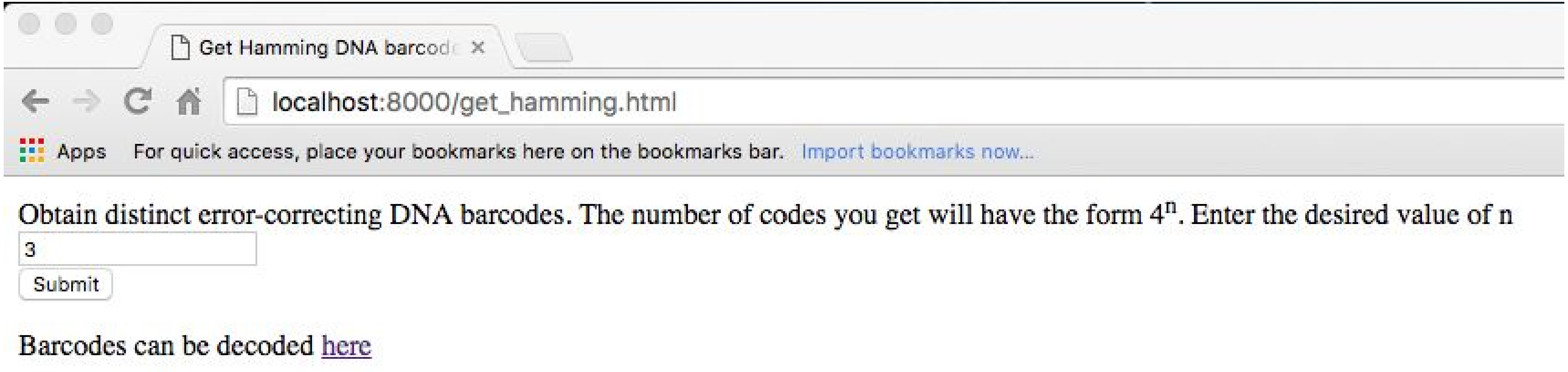
Web interface

Done in Python, making dynamic .cgi web pages and testing using

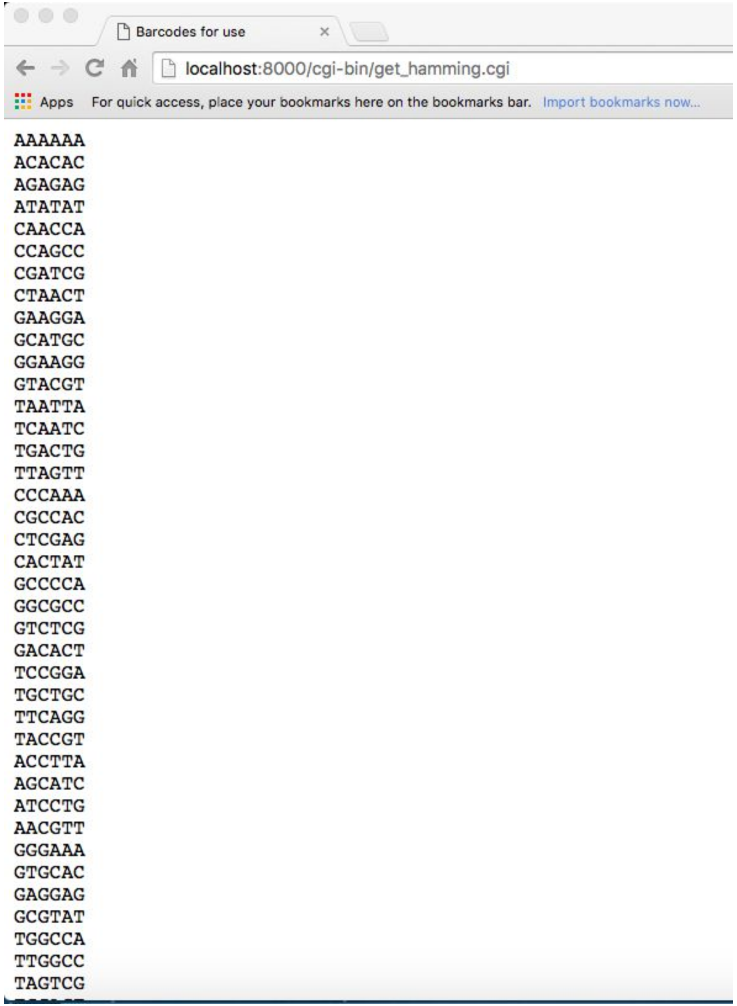
```
python -m CGIHTTPServer
```

```
gershwin|dna/server  python -m CGIHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
127.0.0.1 - - [16/Jun/2016 09:30:09] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [16/Jun/2016 09:30:09] code 404, message File not found
127.0.0.1 - - [16/Jun/2016 09:30:09] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [16/Jun/2016 09:30:13] "GET /decode_hadamard.html HTTP/1.1" 200 -
127.0.0.1 - - [16/Jun/2016 09:32:18] "POST /cgi-bin/decode_hadamard.cgi HTTP/1.1" 200 -
127.0.0.1 - - [16/Jun/2016 09:32:18] CGI script exit status 0x100
127.0.0.1 - - [16/Jun/2016 09:32:51] "POST /cgi-bin/decode_hadamard.cgi HTTP/1.1" 200 -
127.0.0.1 - - [16/Jun/2016 09:34:21] "POST /cgi-bin/decode_hadamard.cgi HTTP/1.1" 200 -
```

web interface - generating Hamming codes



web interface - generating Hamming codes


Barcodes for use

localhost:8000/cgi-bin/get_hamming.cgi

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

AAAAAA
ACACAC
AGAGAG
ATATAT
CAACCA
CCAGCC
CGATCG
CTAACT
GAAGGA
GCATGC
GGAAGG
GTACGT
TAATTA
TCAATC
TGACTG
TTAGTT
CCCAAA
CGCCAC
CTCGAG
CACTAT
GCCCCA
GGCGCC
GTCTCG
GACACT
TCCGGA
TGCTGC
TTCAGG
TACCGT
ACCTTA
AGCATC
ATCCTG
AACGTT
GGGAAA
GTGCAC
GAGGAG
GCGTAT
TGGCCA
TTGGCC
TAGTCG

web interface - decoding Hamming codes

Decode Hamming DNA bar

localhost:8000/decode_hamming.html

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

Enter barcodes to decode, separated by newlines. Output will be a decimal number from 1 to 4^n for each barcode

```
AGAAAA
AAACAC
AGAGAT
ATATAA
CATCCA
CCAGCG
CGATCC
CTACCT
AAAGGA
GGATGC
GGAAGA
GTGCGT
CAATTA
TCAGTC
TTACTG
TTATTT
CCCAAT
CTCCAC
CTCGAC
CACTAC
GCCCCG
GGCGCG
GTCTCT
GACACA
TTCGGA
AGCTGC
GTCAGG
TAGCGT
ACCCTA
```

Submit

Barcodes can be obtained here

web interface - decoding Hamming codes

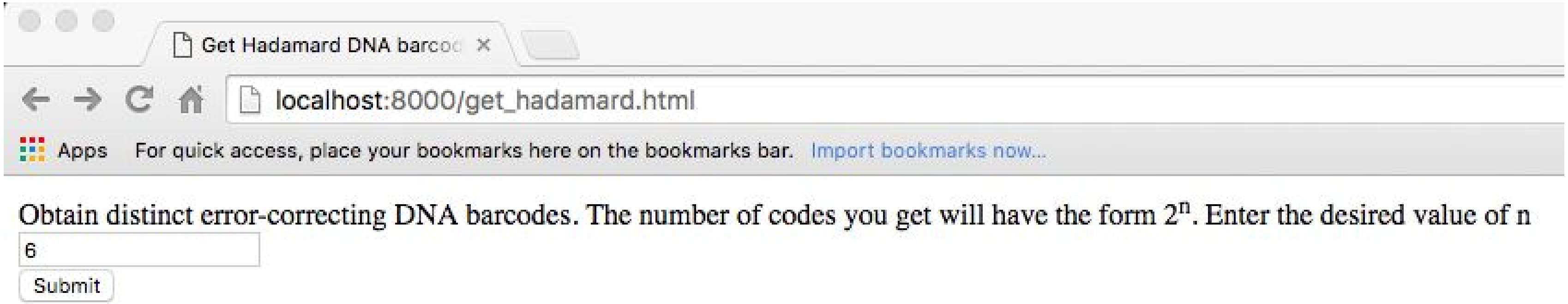
Decoded barcodes

localhost:8000/cgi-bin/decode_hamming.cgi

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now.

```
AGAAAA -> 0
AAACAC -> 1
AGAGAT -> 2
ATATAA -> 3
CATCCA -> 4
CCAGCG -> 5
CGATCC -> 6
CTACCT -> 7
AAAGGA -> 8
GGATGC -> 9
GGAAGA -> 10
GTGCGT -> 11
CAATTA -> 12
TCAGTC -> 13
TTACTG -> 14
TTATTT -> 15
CCCAAT -> 16
CTCCAC -> 17
CTCGAC -> 18
CACTAC -> 19
GCCCCG -> 20
GGCGCG -> 21
GTCTCT -> 22
GACACA -> 23
TTCGGA -> 24
AGCTGC -> 25
GTCAGG -> 26
TAGCGT -> 27
ACCCTA -> 28
AGCCTC -> 29
ATCCAG -> 30
AACGCT -> 31
GGGACA -> 32
GTGCTC -> 33
GAGGAA -> 34
GTGTAT -> 35
TGACCA -> 36
TGGGCC -> 37
GAGTCG -> 38
```

web interface - generating Hadamard codes



web interface - generating Hadamard codes

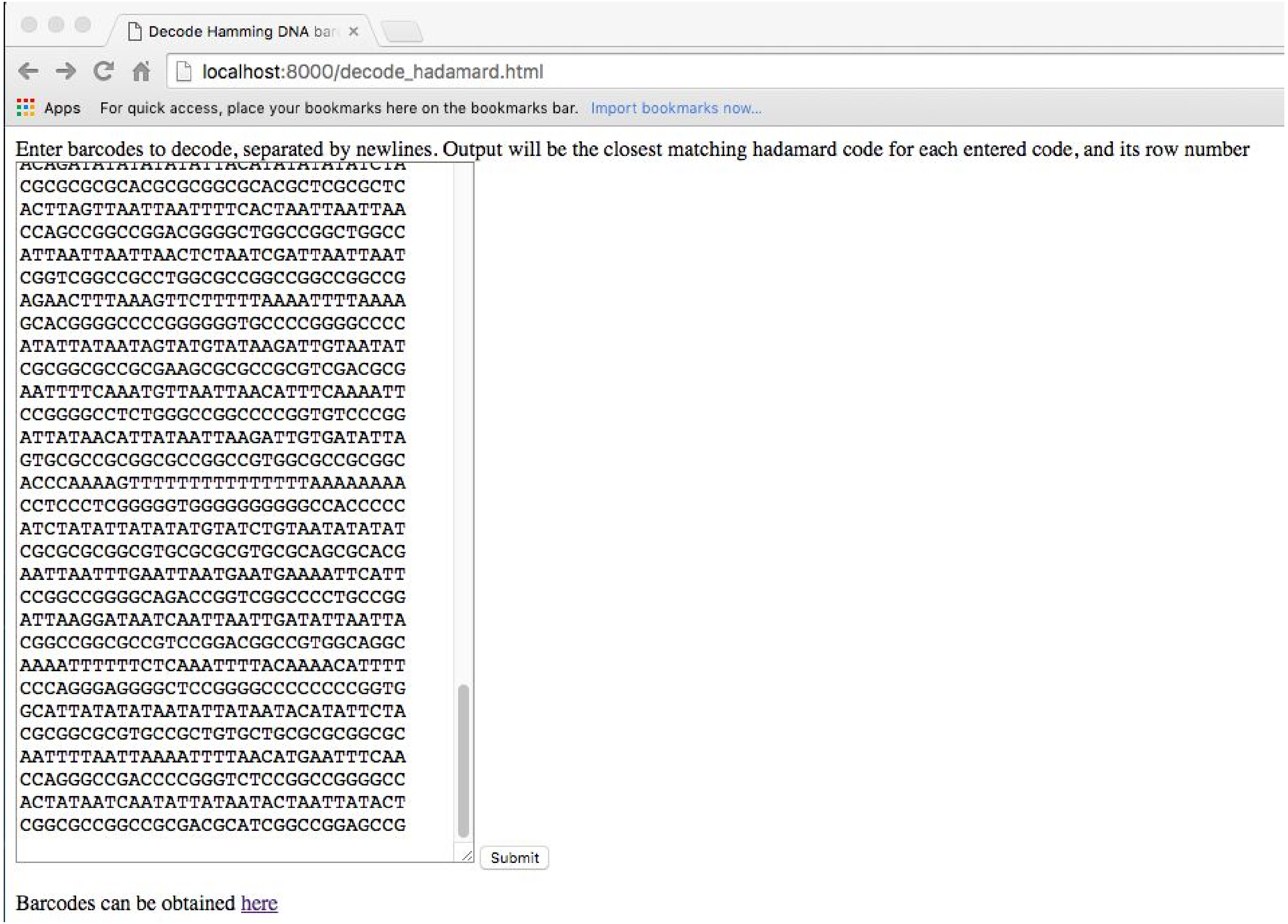
Barcodes for use

localhost:8000/cgi-bin/get_hadamard.cgi

Apps For quick access, place your bookmarks here on the bookmarks bar.

```
TTTTTTTTTTTTTTTT
GGGGGGGGGGGGGGGG
TATATATATATATATA
GCGCGCGCGCGCGCGC
TTAATTAATTAATTAA
GGCCGGCCGGCCGGCC
TAATTAATTAATTAAT
GCCGGCCGGCCGGCCG
TTTTAAAATTTTAAAA
GGGGCCCCGGGGCCCC
TATAATATTATAATAT
GCGCCGCGGCGCCGCG
TTAAAATTTTAAAATT
GGCCCCGGGGCCCCGG
TAATATTATAATATTA
GCCGCGGCGCCGCGGC
TTTTTTTTAAAAAAAA
GGGGGGGGCCCCCCCC
TATATATAATATATAT
GCGCGCGCCGCGCGCG
TTAATTAAAATTAATT
GGCCGGCCCCGGCCGG
TAATTAATATTAATTA
GCCGGCCGCGGCCGGC
TTTTAAAAAAAATTTT
GGGGCCCCCCCCGGGG
TATAATATATATTATA
GCGCCGCGCGCGGCGC
TTAAAATTAATTTTAA
GGCCCCGGCCGGGGCC
TAATATTAATTATAAT
GCCGCGGCCGGCGCCG
AAAAAAAAAAAAAAAA
CCCCCCCCCCCCCCCC
ATATATATATATATAT
CGCGCGCGCGCGCGCG
AATTAATTAATTAATT
CCGGCCGGCCGGCCGG
ATTAATTAATTAATTA
```

Web interface - decoding Hadamard codes



Web interface - decoding Hadamard codes

Decoded barcodes

localhost:8000/cgi-bin/decode_hadamard.cgi

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

Toggle display of distance

```
TTCTTTTTTTTTCTTTTTTTTTTCTTTTTTTT -> TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT (000)
GGGGGGGGTGGGGGGGGGGGGGGGTTGAGGG -> GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG (001)
CATATATATATATATCTATATATATCTATGTA -> TATATATATATATATATATATATATATATATA (002)
GCGCGCGCACGCGCGCTAGCGCGCGCGTGCGC -> GCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC (003)
CTAATTAATCAATCAGTTAATTAATTAATTAA -> TTAATTAATTAATTAATTAATTAATTAATTAA (004)
GGCCGGCCGGCAGGCCGGACGGCAGGCAGGCC -> GGCCGGCCGGCCGGCCGGCCGGCCGGCCGGCC (005)
TAATTAATTAATGAACTGATTAATTGATTAAT -> TAATTAATTAATTAATTAATTAATTAATTAAT (006)
TCCGGCCGGCCTTCCGGCCGGCCGACCGGCCG -> GCCGGCCGGCCGGCCGGCCGGCCGGCCGGCCG (007)
TTGTAAAATTTTAAAATTTTAACATTTCAACA -> TTTTAAAATTTTAAAATTTTAAAATTTTAAAA (008)
GGGGCCCCGGGGCAATGGGGCCCCGGGGCTCC -> GGGGCCCCGGGGCCCCGGGGCCCCGGGGCCCC (009)
TATACTATTATAATGTGATAAGATTATAATAT -> TATAATATTATAATATTATAATATTATAATAT (010)
GCGCCGTGTAGCCGCGGAGCCGCGGCGCCGCG -> GCGCCGCGGCGCCGCGGCGCCGCGGCGCCGCG (011)
TTCAAATTTTAAAATTTCAAAAGTCTAAAATT -> TTAAAATTTTAAAATTTTAAAATTTTAAAATT (012)
GACACCGGGGCCCCGGGGACTCGGGGCCCCGG -> GGCCCCGGGGCCCCGGGGCCCCGGGGCCCCGG (013)
TAATGTCATACCATTATAATATTATAATATTA -> TAATATTATAATATTATAATATTATAATATTA (014)
GCCGCGGTGCCGCGGCGACGCGGTGCCGCTGC -> GCCGCGGCGCCGCGGCGCCGCGGCGCCGCGGC (015)
TTTTGCTTAAAACAAATTTTTCTTAAAAAAAA -> TTTTTTTTAAAAAAAATTTTTTTTAAAAAAAA (016)
GGGGGGGGCCCCCCCAGGGGGGTGCCTCCCTC -> GGGGGGGGCCCCCCCCGGGGGGGGCCCCCCCC (017)
TATATATCACATATAGTATATATAATACATAT -> TATATATAATATATATTATATATAATATATAT (018)
GCGCGCGCCGCTTACGGCGCGCGCCGCGCTCG -> GCGCGCGCCGCGCGCGGCGCGCGCCGCGCGCG (019)
TTAATTAAACTTATTTTTAATTAAAATTAACT -> TTAATTAAAATTAATTTTAATTAAAATTAATT (020)
GACCGGCCCCGGCCGGGGCCGGACCCGGCTGG -> GGCCGGCCCCGGCCGGGGCCGGCCCCGGCCGG (021)
TAATTGATATTAATTATAAGTAATATTAATTG -> TAATTAATATTAATTATAATTAATATTAATTA (022)
GCCGGTCGCGGCCGGCGCAGTCCGCGGCCTGC -> GCCGGCCGCGGCCGGCGCCGGCCGCGGCCGGC (023)
TTTTAACAAAAATTCTTTTTAAAAAAAAGTTT -> TTTTAAAAAAAATTTTTTTTAAAAAAAATTTT (024)
GGAGCCCCCCCCGGGGGGGGCCCCCACAGGTG -> GGGGCCCCCCCCGGGGGGGGCCCCCCCCGGGG (025)
TACAATATAGATTATATATAATATATATTCCA -> TATAATATATATTATATATAATATATATTATA (026)
GCGCAGCGCGCGGAGCGCGCCGCGCTCAGCGC -> GCGCCGCGCGCGGCGCGCGCCGCGCGCGGCGC (027)
TTAAAAGTAATTTTGATTCAAATTCATTTTAA -> TTAAAATTAATTTTAATTAAAATTAATTTTAA (028)
GGCCCCGGCTGTGGCCGGCCCCGGCTGGAGCC -> GGCCCCGGCCGGGGCCGGCCCCGGCCGGGGCC (029)
TGATATTAATGATAGGTAATATTAATTATAAT -> TAATATTAATTATAATTAATATTAATTATAAT (030)
GTAGCGGCCGGCGCCGGACGCGGCCTGCGCCG -> GCCGCGGCCGGCGCCGGCCGCGGCCGGCGCCG (031)
TTTTTTTGTTTTTCTTAAAAAAAAGAAAAAAG -> TTTTTTTTTTTTTTTTAAAAAAAAAAAAAAAA (032)
GGGGAAGGTGGGGGGGCCCCCCCCCCCCCCAC -> GGGGGGGGGGGGGGGGCCCCCCCCCCCCCCCC (033)
TATATAGATATATATAATATATATATATGCAC -> TATATATATATATATAATATATATATATATAT (034)
ACGCGCGCGCGCGCGCAGCTCGCGCGCGCGTG -> GCGCGCGCGCGCGCGCCGCGCGCGCGCGCGCG (035)
TTCATTAATTACTTAAAATTAATGAATTCATT -> TTAATTAATTAATTAAAATTAATTAATTAATT (036)
GGCCGGCCGGCCGGCTCCGGCCGGCACGCCGG -> GGCCGGCCGGCCGGCCCCGGCCGGCCGGCCGG (037)
```

Web interface - decoding Hadamard codes

Decoded barcodes

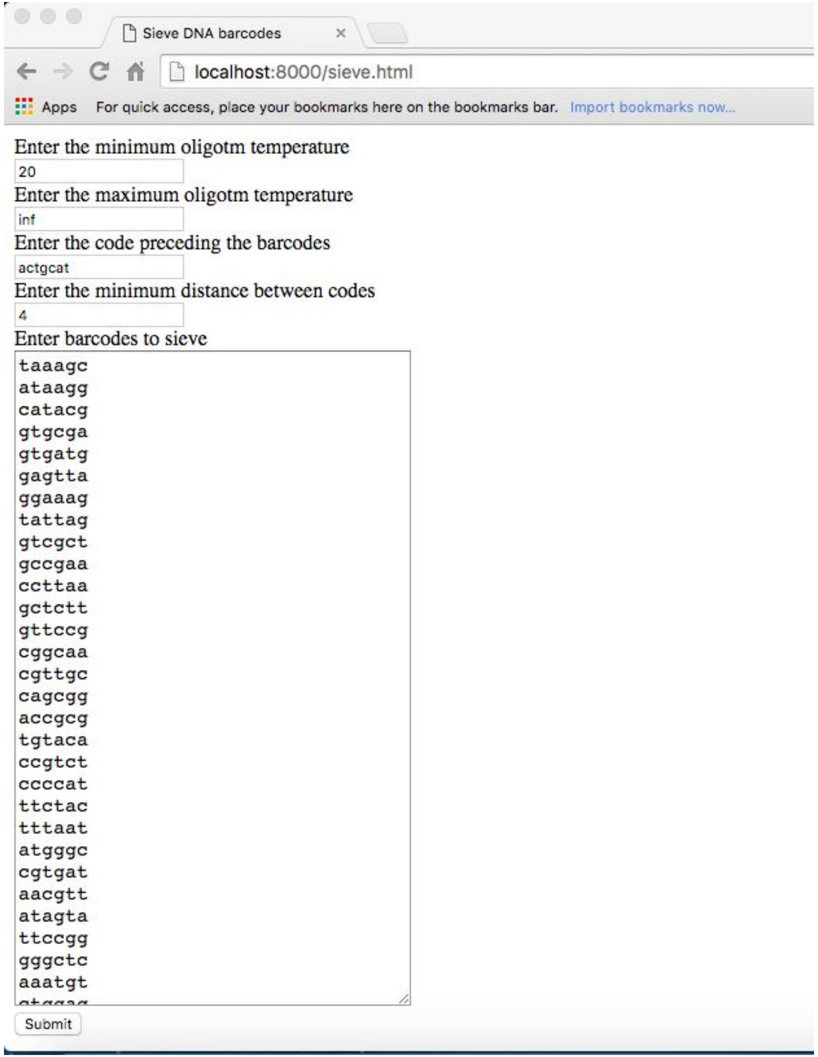
localhost:8000/cgi-bin/decode_hadamard.cgi

Apps For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...

Toggle display of distance

```
TTCTTTTTTTTTCTTTTTTTTTTCTTTTTTTT -> TTTTTTTTTTTTTTTTTTTTTTTTTTTTTTTT (000) distance of 3
GGGGGGGTGGGGGGGGGGGGGGGTTGAGGG -> GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG (001) distance of 4
CATATATATATATATCTATATATATCTATGTA -> TATATATATATATATATATATATATATATATA (002) distance of 4
GCGCGCGCACGCGCGCTAGCGCGCGCGTGCGC -> GCGCGCGCGCGCGCGCGCGCGCGCGCGCGCGC (003) distance of 4
CTAATTAATCAATCAGTTAATTAATTAATTAA -> TTAATTAATTAATTAATTAATTAATTAATTAA (004) distance of 4
GGCCGGCCGGCAGGCCGGACGGCAGGCAGGCC -> GGCCGGCCGGCCGGCCGGCCGGCCGGCCGGCC (005) distance of 4
TAATTAATTAATGAACTGATTAATTGATTAAT -> TAATTAATTAATTAATTAATTAATTAATTAAT (006) distance of 4
TCCGGCCGGCCTTCCGGCCGGCCGACCGGCCG -> GCCGGCCGGCCGGCCGGCCGGCCGGCCGGCCG (007) distance of 4
TTGTAAAATTTTAAAATTTTAACATTTCAACA -> TTTTAAAATTTTAAAATTTTAAAATTTTAAAA (008) distance of 4
GGGGCCCCGGGGCAATGGGGCCCCGGGGCTCC -> GGGGCCCCGGGGCCCCGGGGCCCCGGGGCCCC (009) distance of 4
TATACTATTATAATGTGATAAGATTATAATAT -> TATAATATTATAATATTATAATATTATAATAT (010) distance of 4
GCGCCGTGTAGCCGCGGAGCCGCGGCGCCGCG -> GCGCCGCGGCGCCGCGGCGCCGCGGCGCCGCG (011) distance of 4
TTCAAATTTTAAAATTTCAAAAGTCTAAAATT -> TTAAAATTTTAAAATTTTAAAATTTTAAAATT (012) distance of 4
GACACCGGGGCCCCGGGGACTCGGGGCCCCGG -> GGCCCCGGGGCCCCGGGGCCCCGGGGCCCCGG (013) distance of 4
TAATGTCATACCATTATAATATTATAATATTA -> TAATATTATAATATTATAATATTATAATATTA (014) distance of 4
GCCGCGGTGCCGCGGCGACGCGGTGCCGCTGC -> GCCGCGGCGCCGCGGCGCCGCGGCGCCGCGGC (015) distance of 4
TTTTGCTTAAAACAAATTTTTCTTAAAAAAAA -> TTTTTTTTAAAAAAAATTTTTTTTAAAAAAAA (016) distance of 4
GGGGGGGGCCCCCCCAGGGGGGGTGCCTCCCTC -> GGGGGGGGCCCCCCCCGGGGGGGGCCCCCCCC (017) distance of 4
TATATATCACATATAGTATATATAATACATAT -> TATATATAATATATATTATATATAATATATAT (018) distance of 4
GCGCGCGCCGCTTACGGCGCGCGCCGCGCTCG -> GCGCGCGCCGCGCGCGGCGCGCGCCGCGCGCG (019) distance of 4
TTAATTAAACTTATTTTTAATTAAAATTAACT -> TTAATTAAAATTAATTTTAATTAAAATTAATT (020) distance of 3
GACCGGCCCCGGCCGGGGCCGGACCCGGCTGG -> GGCCGGCCCCGGCCGGGGCCGGCCCCGGCCGG (021) distance of 3
TAATTGATATTAATTATAAGTAATATTAATTG -> TAATTAATATTAATTATAATTAATATTAATTA (022) distance of 3
GCCGGTCGCGGCCGGCGCAGTCCGCGGCCTGC -> GCCGGCCGCGGCCGGCGCCGGCCGCGGCCGGC (023) distance of 4
TTTTAACAAAAATTCTTTTTAAAAAAAAGTTT -> TTTTAAAAAAAATTTTTTTTAAAAAAAATTTT (024) distance of 3
GGAGCCCCCCCCGGGGGGGGCCCCCACAGGTG -> GGGGCCCCCCCCGGGGGGGGCCCCCCCCGGGG (025) distance of 4
TACAATATAGATTATATATAATATATATTCCA -> TATAATATATATTATATATAATATATATTATA (026) distance of 4
GCGCAGCGCGCGGAGCGCGCCGCGCTCAGCGC -> GCGCCGCGCGCGGCGCGCGCCGCGCGCGGCGC (027) distance of 4
TTAAAAGTAATTTTGATTCAAATTCATTTTAA -> TTAAAATTAATTTTAATTAAAATTAATTTTAA (028) distance of 4
GGCCCCGGCTGTGGCCGGCCCCGGCTGGAGCC -> GGCCCCGGCCGGGGCCGGCCCCGGCCGGGGCC (029) distance of 4
TGATATTAATGATAGGTAATATTAATTATAAT -> TAATATTAATTATAATTAATATTAATTATAAT (030) distance of 4
GTAGCGGCCGGCGCCGGACGCGGCCTGCGCCG -> GCCGCGGCCGGCGCCGGCCGCGGCCGGCGCCG (031) distance of 4
TTTTTTTGTTTTCTTAAAAAAAAGAAAAAAG -> TTTTTTTTTTTTTTTTAAAAAAAAAAAAAAAA (032) distance of 4
GGGGAAGGTGGGGGGGCCCCCCCCCCCCCCAC -> GGGGGGGGGGGGGGGGCCCCCCCCCCCCCCCC (033) distance of 4
TATATAGATATATATAATATATATATATGCAC -> TATATATATATATATAATATATATATATATAT (034) distance of 4
ACGCGCGCGCGCGCGCAGCTCGCGCGCGCGTG -> GCGCGCGCGCGCGCGCCGCGCGCGCGCGCGCG (035) distance of 4
TTCATTAATTACTTAAAATTAATGAATTCATT -> TTAATTAATTAATTAAAATTAATTAATTAATT (036) distance of 4
GGCCGGCCGGCCGGCTCCGGCCGGCACGCCGG -> GGCCGGCCGGCCGGCCCCGGCCGGCCGGCCGG (037) distance of 3
```

web interface - sieving



web interface - sieving