Code Golf

# Introduction

‘Code Golf’ is so-called because the idea is to use as few ‘shots’ (characters) as possible. Try to make your code as short as possible, making use of language features that help with this.

Once the competition is over, consider posting the code to GitHub (or Twitter if it’s short enough) so that other coders can learn how your code works. This is not mandatory, but is polite and encourages others to code. I for one would love to learn more about other golfers’ languages.

Best of luck.

# Rules

* The code must compile/run and give the correct output for any given test cases.
* The code must be ready to demonstrate at 8:30pm. If you have time, code some/all test cases in a separate file showing that it works for everything.

# Hole 1: Simple

Given a single positive integer *n*, output the decimal (base 10) representation of *n* when its most significant bit is set to 0 (zero).

## Test Cases

1 -> 0

2 -> 0

10 -> 2

16 -> 0

100 -> 36

267 -> 11

350 -> 94

500 -> 244

## Worked Example

350 in binary is **1**01011110.

Setting its most significant bit (i.e. the leftmost 1 bit) to 0 turns it into **0**01011110.

This is equivalent to the decimal integer 94.

# Hole 2: Medium

Given the letter A, B or C and a number 0-10, output the size of the matching standard paper size (Series A and B) or the matching standard envelope size (series C) in millimetres, in the format *wwww x hhhh* where *wwww* and *hhhh* are the width and height measurements in millimetres as per ISO216 (Series A & B) or ISO296 (Series C). The measurements are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| **Format** | **A** | **B** | **C** |
| **Size** | **mm × mm** | **mm × mm** | **mm × mm** |
| **0** | 841 × 1189 | 1000 × 1414 | 917 × 1297 |
| **1** | 594 × 841 | 707 × 1000 | 648 × 917 |
| **2** | 420 × 594 | 500 × 707 | 458 × 648 |
| **3** | 297 × 420 | 353 × 500 | 324 × 458 |
| **4** | 210 × 297 | 250 × 353 | 229 × 324 |
| **5** | 148 × 210 | 176 × 250 | 162 × 229 |
| **6** | 105 × 148 | 125 × 176 | 114 × 162 |
| **7** | 74 × 105 | 88 × 125 | 81 × 114 |
| **8** | 52 × 74 | 62 × 88 | 57 × 81 |
| **9** | 37 × 52 | 44 × 62 | 40 × 57 |
| **10** | 26 × 37 | 31 × 44 | 28 × 40 |

## Test Cases

|  |  |  |
| --- | --- | --- |
| \*\*Test case 1\*\*  Input: A4  Output: 210 x 297 | \*\*Test Case 2\*\*  Input: B6  Output: 125 x 176 | \*\*Test Case 3\*\*  Input: C2  Output: 458 x 648 |

## Notes

* Whilst the format "210 x 297" or "1000 x 1414"is preferable, you can choose to omit the "×" from your output and instead return it in the form of an array of two numbers. In fact do whatever tickles your fancy as long as the width is outputted before the height.
* In successive sizes for a series as you go down the table, the width for one size becomes the height for the next. This may also help you golf down your code.
* The ratio between the height and the width is roughly equivalent to the square root of 2, so in the calculation of the heights, the width is multiplied by , and then rounded up or down to the nearest millimetre, thus resulting in the measurements in the table above. This may help golf down your code. **This is true for *most* values but not *all*. Tread carefully.**

# Hole 3: Hard

Given an input of one of more words separated by spaces, sort the words alphabetically and print them out vertically in 3 columns separated by spaces.

All three columns’ heights should be left aligned, and as evenly weighted as possible

## Test Cases

Input: “a b c d e f g”

Output:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| a c e  b d f   g | or | a d f  b e g  c | but NOT | a d g  b e  c f |

Input: “cat caterpillar pie frog elephant pizza”

Output:

cat elephant pie

caterpillar frog pizza

Input: “hello this is just me trying to test a long sentence and see what happens”

Output:

a hello me

and is see

application just test

happens long this

# Answers (in Python)

## Hole 1

print int('0'+bin(raw\_input())[3:],2)

## Hole 2

a,b,c,r =26,31,28,1.4142

m,n=[209,840,124,249,161,916,499],[115,177,354]

s={'A':26,'B':31,'C':28}

def ht(w):

return [w,int(round(r\*w))]

x=raw\_input()

w,h=ht(s[x[0]])

for i in range(10-int(x[1:])):

w,h=ht(h)

h=h+1 if h in m else h-1 if h in n else h

print w,h

## Hole 3

i=raw\_input().split(' ')

i.sort()

if len(i)>3:

n,o,s,t,v,z=len(i),[],[],0,[],[0,1,2]

r=((n-1)/3)+1

for c in z:

v+=[i[t:t+r-1+int(c<(n%3 if n%3>0 else 3))]]

k=v[c]

t+=len(k)

s+=[max([len(w) for w in k])]

if len(k)<r:k+=[' '\*s[c]]

for l in range(r):o+=[v[i][l]+' '\*(s[i]+1-len(v[i][l])) for i in z]+['\n']

print ''.join(o)

else:

print ' '.join(i)