Contents

Chapter 1

Functions

1.1 arygcd – binary-like gcd algorithms

1.1.1 bit_num – the number of bits

bit_num(`a`: *integer*) → *integer*

Return the number of bits for `a`

1.1.2 binarygcd – gcd by the binary algorithm

binarygcd(`a`: *integer*, `b`: *integer*) → *integer*

Return the greatest common divisor (gcd) of two integers `a`, `b` by the binary gcd algorithm.

1.1.3 arygcd_i – gcd over gauss-integer

arygcd_i(`a1`: *integer*, `a2`: *integer*, `b1`: *integer*, `b2`: *integer*)
→ (*integer*, *integer*)

Return the greatest common divisor (gcd) of two gauss-integers `a1`+`a2`i, `b1`+`b2`i, where "i" denotes the imaginary unit.

If the output of arygcd_i(`a1`, `a2`, `b1`, `b2`) is (`c1`, `c2`), then the gcd of `a1`+`a2`i and `b1`+`b2`i equals `c1`+`c2`i.
†This function uses $(1 + i)$-ary gcd algorithm, which is an generalization of the binary algorithm, proposed by A.Weilert[2].

1.1.4 arygcd_w – gcd over Eisenstein-integer

arygcd_w(`a1`: *integer*, `a2`: *integer*, `b1`: *integer*, `b2`: *integer*)
→ (*integer*, *integer*)

Return the greatest common divisor (gcd) of two Eisenstein-integers `a1`+`a2`ω, `b1`+`b2`ω, where "ω" denotes a primitive cubic root of unity.

If the output of arygcd_w(`a1`, `a2`, `b1`, `b2`) is (`c1`, `c2`), then the gcd of `a1`+`a2`ω and `b1`+`b2`ω equals `c1`+`c2`ω.
†This functions uses $(1-\omega)$-ary gcd algorithm, which is an generalization of the binary algorithm, proposed by I.B. Damgård and G.S. Frandsen [1].

Examples

```
>>> arygcd.binarygcd(32, 48)
16
>>> arygcd_i(1, 13, 13, 9)
(-3, 1)
>>> arygcd_w(2, 13, 33, 15)
(4, 5)
```

Bibliography

[1] Ivan Bjerre Damgård and Gudmund Skovbjerg Frandsen. Efficient algorithms for the gcd and cubic residuosity in the ring of Eisenstein integers. *Journal of Symbolic Computation*, Vol. 39, No. 6, pp. 643–652, 2005.

[2] André Weiler. $(1+i)$-ary gcd computation in $\mathbb{Z}[i]$ as an analogue to the binary gcd algorithm. *Journal of Symbolic Computation*, Vol. 30, No. 5, pp. 605–617, 2000.