

Contents

1	Functions	2
1.1	arith1 - miscellaneous arithmetic functions	2
1.1.1	floorsqrt - floor of square root	2
1.1.2	floorpowerroot - floor of some power root	2
1.1.3	legendre - Legendre(Jacobi) Symbol	2
1.1.4	modsqrt - square root of a for modulo p	2
1.1.5	expand - m -adic expansion	3
1.1.6	inverse - inverse	3
1.1.7	CRT - Chinese Remainder Theorem	3
1.1.8	AGM - Arithmetic Geometric Mean	3
1.1.9	vp - p -adic valuation	3
1.1.10	issquare - Is it square?	4
1.1.11	log - integer part of logarithm	4
1.1.12	product - product of some numbers	4

Chapter 1

Functions

1.1 arith1 - miscellaneous arithmetic functions

1.1.1 floorsqrt – floor of square root

`floorsqrt(a: integer/Rational) → integer`

Return the floor of square root of a.

1.1.2 floorpowerroot – floor of some power root

`floorpowerroot(n: integer, k: integer) → integer`

Return the floor of k-th power root of n.

1.1.3 legendre - Legendre(Jacobi) Symbol

`legendre(a: integer, m: integer) → integer`

Return the Legendre symbol or Jacobi symbol $\left(\frac{a}{m}\right)$.

1.1.4 modsqrt – square root of a for modulo p

`modsqrt(a: integer, p: integer) → integer`

Return one of the square roots of a for modulo p if square roots exist, raise ValueError otherwise.

p must be a prime number.

1.1.5 expand – m -adic expansion

expand(n : *integer*, m : *integer*) → *list*

Return the m -adic expansion of n .

n must be nonnegative integer. m must be greater than or equal to 2. The output is a list of expansion coefficients in ascending order.

1.1.6 inverse – inverse

inverse(x : *integer*, n : *integer*) → *integer*

Return the inverse of x for modulo n .

n must be coprime to x .

1.1.7 CRT – Chinese Remainder Theorem

CRT($nlist$: *list*) → *integer*

Return the uniquely determined integer satisfying all modulus conditions given by $nlist$.

Input list $nlist$ must be the list of a list consisting of two elements. The first element is remainder and the second is divisor. They must be integer.

1.1.8 AGM – Arithmetic Geometric Mean

AGM(a : *integer*, b : *integer*) → *float*

Return the Arithmetic-Geometric Mean of a and b .

1.1.9 vp – p -adic valuation

vp(n : *integer*, p : *integer*, k : *integer=0*) → *tuple*

Return the p -adic valuation and other part for n .

†If k is given, return the valuation and the other part for np^k .

1.1.10 issquare - Is it square?

`issquare(n: integer) → integer`

Check if n is a square number and return square root of n if n is a square. Otherwise, return 0.

1.1.11 log – integer part of logarithm

`log(n: integer, base: integer=2) → integer`

Return the integer part of logarithm of n to the base.

1.1.12 product – product of some numbers

`product(iterable: list, init: integer/Rational=None)
→ prod: integer/Rational`

Return the products of all elements in iterable.

If `init` is given, the multiplication starts with `init` instead of the first element in iterable.

Input list `iterable` must be list of numbers including integers, **Rational** etc. The output `prod` may be determined by the type of elements of `iterable` and `init`.

Examples

```
>>> arith1.AGM(10, 15)
12.373402181181522
>>> arith1.CRT([[2, 5], [3, 7]])
17
>>> arith1.CRT([[2, 5], [3, 7], [5, 11]])
192
>>> arith1.expand(194, 5)
[4, 3, 2, 1]
>>> arith1.vp(54, 3)
(3, 2)
```

```
>>> arith1.product([1.5, 2, 2.5])
7.5
>>> arith1.product([3, 4], 2)
24
>>> arith1.product([])
1
```

Bibliography