In [9]:

```
import sympy as sym
x = sym.Symbol('x')
equation = (3*x) - 4 + 2
exact_solutions = sym.solvers.solve(equation, x)
print("Solutions:", exact_solutions[0])
only_solution = exact_solutions[0]
print("Solution as a floating point number:", only_solution.evalf())
```

```
Solutions: 2/3
Solution as a floating point number: 0.666666666666667
```

In [17]:

```
from sympy import sieve,prime
print([i for i in sieve.primerange(19)])
```

```
[2, 3, 5, 7, 11, 13, 17]
```

In [18]:

```
print([i for i in sieve.primerange(7,19)])
```

```
[7, 11, 13, 17]
```

In [19]:

```
list(sieve.primerange(prime(10)+1))
```

Out[19]:

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
```

In [20]:

```
from sympy import prime
prime(1)
```

Out[20]:

```
2
```

In [21]:

```
prime(10)
```

Out[21]:

```
29
```

In [23]:

```
from sympy import Q,pi,ask
from sympy.abc import x,y
ask(Q.rational(pi))
```

Out[23]:

```
False
```

In [26]:

```
ask((Q.even(x*y)),(Q.even(x))&(Q.even(y)))
```

Out[26]:

```
True
```

In [29]:

```
ask(Q.positive(x),x>0)
```

In [30]:

```
Q.integer(1)
```

Out[30]:

$Q_{\mathrm{integer}}(1)$

In [33]:

```
from sympy.stats import density, MultivariateT
from sympy import Symbol, pprint
x = Symbol("x")
X = MultivariateT("x", [1, 1], [[1, 0], [0, 1]], 2)
multiVar = density(X)(1, 2)
pprint(multiVar)
```

```
 2 
───
9⋅π
```

In [2]:

```
from sympy.combinatorics import Partition
a=Partition([1,2],[3],[4],[5,6])
```

In [5]:

```
a.members
```

Out[5]:

```
(1, 2, 3, 4, 5, 6)
```

In [6]:

```
a.RGS
```

Out[6]:

```
(0, 0, 1, 2, 3, 3)
```

In [7]:

```
a+1
```

Out[7]:

$$\{\{3\},\{4\},\{5\},\{6\},\{1,2\}\}$$

In [8]:

```
_.RGS
```

Out[8]:

```
(0, 0, 1, 2, 3, 4)
```

In [9]:

```
Partition([1],[2,3],[4]).partition
```

Out[9]:

```
[[1], [2, 3], [4]]
```

In [11]:

```
a=Partition([1,2],[3],[4,5])
a.rank
```

Out[11]:

$$13$$

In [15]:

```
from sympy.combinatorics.partitions import IntegerPartition
IntegerPartition([1]*3+[2]+[4]*5).as_dict()
```

Out[15]:

```
{4: 5, 2: 1, 1: 3}
```

In [18]:

```
print(IntegerPartition([1,2,3,4,5,6,7,8,9]).as_ferrers())
```

```
#########
########
#######
######
#####
####
###
##
#
```

In [19]:

```
from sympy import *

a = Rational(5, 8)
print("value of a is :" + str(a))

b = Integer(5.579)
print("value of b is :" + str(b))
```

```
value of a is :5/8
value of b is :5
```

In [20]:

```
from sympy import *


p = pi**3
print("value of p is :" + str(p))


q = pi.evalf()
print("value of q is :" + str(q))


r = exp(1).evalf()
print("value of r is :" + str(r))

s = (pi + exp(1)).evalf()
print("value of s is :" + str(s))

rslt = oo + 10000
print("value of rslt is :" + str(rslt))

if oo > 9999999 :
    print("True")
else:
    print("False")
```

```
value of p is :pi**3
value of q is :3.14159265358979
value of r is :2.71828182845905
value of s is :5.85987448204884
value of rslt is :oo
True
```

In [21]:

```
from sympy import *
x = Symbol('x')
y = Symbol('y')

z = (x + y) + (x-y)
print("value of z is :" + str(z))
```

```
value of z is :2*x
```

In [22]:

```
from sympy import *


x = Symbol('x')
ans1 = diff(sin(x)*exp(x), x)
print("derivative of sin(x)*e ^ x : ", ans1)

ans2 = integrate(exp(x)*sin(x) + exp(x)*cos(x), x)
print("indefinite integration is : ", ans2)


ans3 = integrate(sin(x**2), (x, -oo, oo))
print("definite integration is : ", ans3)


ans4 = limit(sin(x)/x, x, 0)
print("limit is : ", ans4)


ans5 = solve(x**2 - 2, x)
print("roots are : ", ans5)
```

```
derivative of sin(x)*e ^ x :  exp(x)*sin(x) + exp(x)*cos(x)
indefinite integration is :  exp(x)*sin(x)
definite integration is :  sqrt(2)*sqrt(pi)/2
limit is :  1
roots are :  [-sqrt(2), sqrt(2)]
```