# Diskret matematik

# Programmeringslaboration

## Kombinatorik och sannolikhet

Jenny Söderberg 890728-6648

Alexander Milton 940510-8136

Ht2014

### Uppgift 1: Tärningar

Räkna ut sannolikheten P(E|F) för n kast med en m-sidig tärning, där

E = ”alla sidor på tärningen är med i följden av tärningstest”,

F = ”följden av tärningskast är växande”.

*a)*

Räkna ut P(E), P(F) och P(E|F) för (m,n) {(6,8),(6,6),(4,5),(4,10),(3,10),(2,10)}

#### Omega

integer n tärningskast

integer m sidor

integer[][] Omega( n, m )

{

integer[][] omega

integer i, j

for( 0 < i < m^n)

{

for( 0 < j < n )

{

omega[i][j] = (i/( m^j ) %m )+1

}

}

return omega

}

#### E

integer n tärningskast

integer m sidor

integer[][] E( n, m )

{

integer[][] e

integer i, die

boolean allValues, set to True

for( 0 < i < omega.size )

{

for( 1 < die <= m )

{

if ( omega[i] does not contain die )

{

allValues = false

}

}

if( allValues = true )

{

put omega[i] in e

}

}

return e

}

#### F

integer n tärningskast

integer m sidor

integer[][] F( n, m )

{

integer n tärningskast

integer m sidor

integer[][] f

integer i, j

boolean lessThan, set to False

for ( 0 < i < omega.size )

{

for ( 1 < j < omega[i].size )

{

if ( omega[i][j] < omega[i][j - 1] )

{

lessThan = true

}

}

if ( lessThan = false )

{

put omega[i] in f

}

}

return f

}

#### Section

integer[][] A, B

integer[][] section( A, B )

{

integer[][] sect

integer i, j

for ( 0 < i < A.size )

{

for ( 0 < j < B.size )

{

if ( A[i] = B[i] )

{

put A[i] in sect

}

}

}

return sect

}

#### Calculate probability 1a

integer n tärningskast

integer m sidor

Input: (m,n) {(6,8),(6,6),(4,5),(4,10),(3,10),(2,10)}

E = E(n,m)

F = F(n,m)

E∩F = section(E,F)

Print “P(E): ” E.size/omega.size

Print “P(F): “ F.size/omega.size

Print “P(E|F): “ E∩F.size/F.size

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **m** | **n** | **P(E)** | **P(F)** | **P(E|F)** |
| 6 | 8 | 0.114026 | 0.000766247 | 0.016317 |
| 6 | 6 | 0.0154321 | 0.00990226 | 0.0021645 |
| 4 | 5 | 0.234375 | 0.0546875 | 0.0714286 |
| 4 | 10 | 0.780602 | 0.000272751 | 0.293706 |
| 3 | 10 | 0.948026 | 0.00111772 | 0.545455 |
| 2 | 10 | 0.998047 | 0.0107422 | 0.818182 |

### Slutsats:

P(E|F) är sannolikheten för att alla tärningsvärden finns med om värdena i följden är stigande.

b)

Undersök den ungefärliga sannolikheten P(E|F) när experimentet görs x gånger.

integer n tärningskast

integer m sidor

integer x gånger

#### Kasta tärning

integer kasta\_tarning( m )

{

if ( m < 2 )

{

return NULL

}

integer value = randomValue % m

return value

}

#### Kasta tärningar n gånger

integer[] kasta\_tarningar( n, m )

{

integer[] values

for ( 0 < i < n )

{

put ( kasta\_tarning( m ) ) in values

}

return values

}

#### Calculate probability 1b

integer n, m, x

oneBprobability( n, m, x )

{

integer[][] values

for ( 0 < i < x )

{

int[] tempValues = kasta\_tarningar( n, m )

put tempValues in values

}

omega = values

integer b = F( n, m ).size

integer a = section( E( n, m ), F( n, m ).size )

if (b ≠ 0)

{

float probability = a / b

print “P(E|F): ” probability

}

}

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **m** | **n** | **X = 1000** | **X = 10^4** | **X=10^5** | **P(E|F)** |
| 6 | 8 | 0 | 0 | 0.0229885 | 0.016317 |
| 6 | 6 | 0 | 0 | 0.00391389 | 0.0021645 |
| 4 | 5 | 0.0444444 | 0.0597826 | 0.0706121 | 0.0714286 |
| 4 | 10 | 0 | 0.333333 | 0.318182 | 0.293706 |
| 3 | 10 | 0 | 0.428571 | 0.5 | 0.545455 |
| 2 | 10 | 0.75 | 0.821053 | 0.840156 | 0.818182 |

c)

För m ≤ n

d)

### Uppgift 2: Binomialkoefficienter

a)

n är radindex i Pascals triangel, k är index för elementet i en rad

#### Binomialkoefficienter

integer n, k

integer binom\_pascal( n, k )

{

if ( k > n )

{

return 0

}

integer[][] pascalVector

for ( 0 < i <= n ) // Row number

{

integer[] rowVector

for ( 0 < j <= i ) // Element in row

{

if ( j = 0 or j = i )

{

put 1 in rowVector

print 1

}

else

{

integer firstValue = pascalVector[i – 1] [j – 1]

integer secondValue = pascalVector[i - 1][j]

print (firstValue + secondValue)

put (firstValue + secondValue) in rowVector

}

}

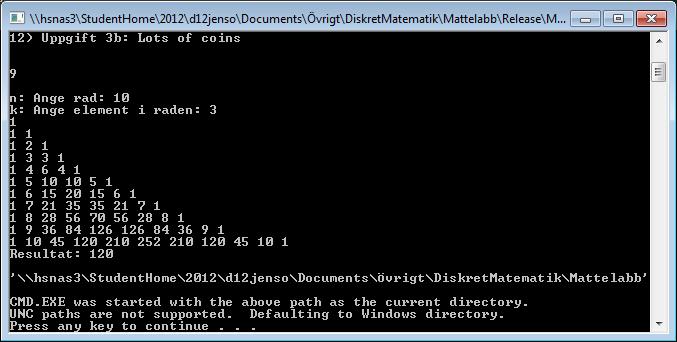
put rowVector in pascalVector

}

return pascalVector[n][k]

}

Funktionen ritar ut Pascals triangel så långt ner som behövs för att nå den önskade raden och returnerar elementet på angiven indexplats. I exemplet blir alltså element 3 det fjärde elementet i raden.



b)

Utveckla (x + y)^n

#### Expand

string x, y

integer n

expansion( x, y, n )

{

for ( 0 < i <= n )

{

integer k = binom\_pascal( n, i )

print: k "(" x ")^" (n – i) " (" y ")^" i

if (i ≠ n)

{

print " + "

}

}

}

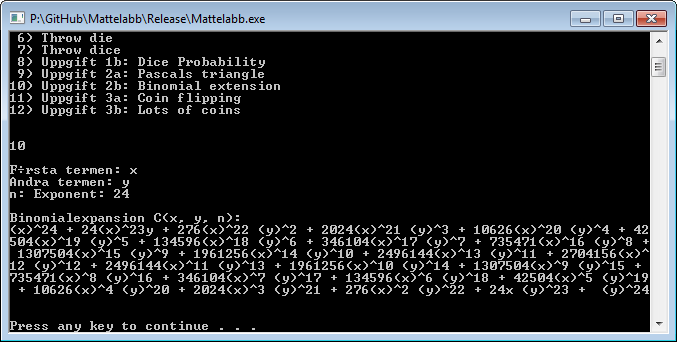
Note: k, i och (n-i) skrivs inte ut om de har värdet 1.

x och y skrivs inte ut om de har värdet 0.

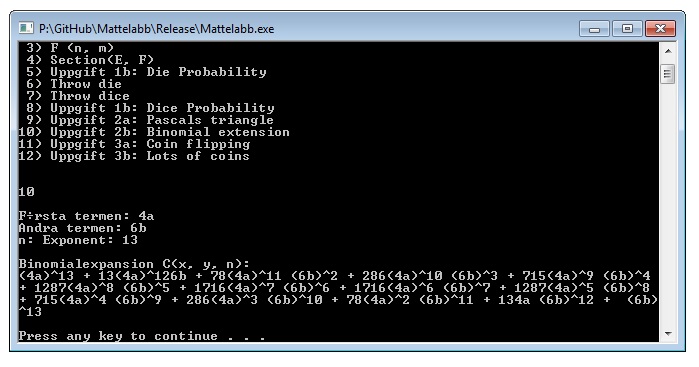
Koefficienterna för varje term fås av binomialsatsen.

c) Utveckla

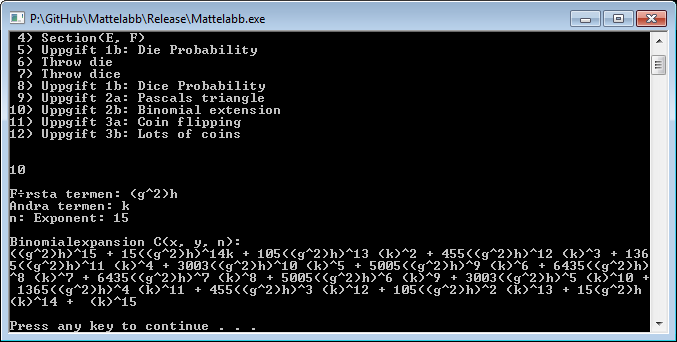
i.



ii.



iii.



### Uppgift 3: Binomialfördelning

a)

krona: 1

klave: 0

#### Kasta mynt

float p

integer kasta\_mynt( p )

{

integer probability = p \* 1000

integer result = randomValue % 1000

if (result < probability)

{

return 1

}

else

{

return 0

}

}

Eftersom det bara finns två möjliga utfall, krona och klave, måste varje resultat som inte är krona vara klave. Det ger att om sannolikheten för krona är p, blir sannolikheten för klave q = 1 – p då de kombinerade sannolikheterna för alla möjliga utfall alltid är ett.

b)

#### Mynt\_experiment

integer n, k

float p

boolean mynt\_experiment( n, k, p )

{

integer count = 0

for ( 0 < i < n )

{

count = ( count + kasta\_mynt( p ) )

}

return true if ( count = k )

}

#### Print\_experiment

integer n

float p

print\_experiment( n, p )

{

integer x = 100000000

for ( 0 < k <= n )

{

integer h = 0

for ( 0 < i < x )

{

if ( mynt\_experiment( n, k, p ) ) returns 1

{

h = h + 1;

}

}

print "k: " k

print "h/x: " h/x

}

}

Syftet var att undersöka sannolikheten för att få krona exakt k gånger när ett mynt kastas n gånger och sannolikheten p för krona i varje enskilt kast är 0.3. Experimentet utfördes 10^8 gånger för varje k<n, för n= 2,3,4,5,10.

n: gånger

k: önskat antal krona

p: sannolikhet för krona

h: antal lyckade experiment

x: antal gånger experimentet utförs

Experimentet utfördes med x=100000000, d.v.s. 10^8 gånger

bool experiment( n, k, p )

h/x för k = 0-n, p = 0.3, n = 2,3,4,5,10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | N = 2 | N=3 | N=4 | N=5 | N=10 |
| K=0 | 0.487056 | 0.339862 | 0.237234 | 0.165521 | 0.0273878 |
| K=1 | 0.421705 | 0.441535 | 0.410788 | 0.358352 | 0.118642 |
| K=2 | 0.0912536 | 0.191092 | 0.266805 | 0.310188 | 0.231196 |
| K=3 | - | 0.0275328 | 0.0769986 | 0.134343 | 0.266817 |
| K=4 | - | - | 0.00833216 | 0.0291038 | 0.202152 |
| K=5 | - | - | - | 0.00251081 | 0.104976 |
| K=6 | - | - | - | - | 0.0378986 |
| K=7 | - | - | - | - | 0.00936076 |
| K=8 | - | - | - | - | 0.00151394 |
| K=9 | - | - | - | - | 0.00014641 |
| K=10 | - | - | - | - | 6.01\*10^-6 |

Eftersom sannolikheten för att få krona är lite mindre än hälften så stor som sannolikheten för att få krona, ( p = 0.3 ger 1 – p = q = 0.7 ), blir det inte en normalfördelning av resultaten. Det som fås är en tvåpunktsfördelning, mer specifikt en Bernoullifördelning då p ≠ q.

c