EXPLORING MATHEMATICS WITH MATHEMATICA

Cheri Shakiban
Professor of Mathematics
University of St. Thomas
St. Paul, Mn. 55105

PART 1- NUMBERS

Chapter 1. Modular Arithmetic.

1.1. Basic Definitions.

When doing certain calculations it turns out that the particular value of some integer x is largely irrelevant and what is important is its remainder after division by some other number.

Definition 1.1. Let d be a positive integer. If x and y are integers where their difference is multiple of d, we say that a and b are *congruent modulo d* and write

$x \equiv y \pmod{d}$.

Example 1.1. We have $123 \equiv 18 \pmod 5$, since 5 divides (123–18) =105. Likewise, $33 \equiv -17 \pmod 5$. On the other hand, 23 and 9 are not congruent mod 5 since 5 does not divide (23–9)=14. Congruences arise in everyday life. For instance, clocks work either modulo 12 or 24 for hours, and modulo 60 for minutes and seconds, calendars work modulo 7 for days of the week and modulo 12 for months. Utility meters often operate modulo 1000, and odometers usually work modulo 100000.

The MATHEMATICA command `Mod[m,d]` gives the remainder of division of *m* by *d*.

```
Mod[3200,47]
```

```
4
```

We can use `Mod` to test if two integers are congruent modulo a positive integer *d*.

```
Mod[123,5]==Mod[18,5]
```

```
True
```

For the numbers of the form a^b, we can also use `PowerMod[a,b,d]` which gives a^b mod d but is more efficient than `Mod[a^b,d]`.

We can add, subtract or multiply any pair of congruences, modulo the same integer d. More precisely if $a \equiv a_1 \pmod d$ and $b \equiv b_1 \pmod d$ then

$a + b \equiv a_1 + b_1 \pmod d$

$a - b \equiv a_1 - b_1 \pmod d$

$ab \equiv a_1 b_1 \pmod d$.

The proof is not very hard and is left as an exercise.

Example 1.2. We have $123 \equiv 18 \pmod 5$ and $33 \equiv -17 \pmod 5$, then

$123+33 \equiv 18-17 \pmod 5$.

It is obvious that the set of integers can be divided into d different sets each containing integers that are mutually congruent modulo d. These sets are called *residue classes modulo d*, and are denoted by [0], [1], [2], ...,[d–1]. Hence every integer is congruent modulo d to exactly one of the numbers 0, 1, 2, ..., d–1, which is the complete set of the possible remainders of division of any integer by d. It is often convenient, when working in modulo d, to represent an integer by its corresponding residue class. For instance modulo d 1234 is represented by [2] , since $1234 \equiv 2 \pmod 4$.

Example 1.3. The four congruence classes modulo 4 are given by

[0] $\ldots \equiv -8 \equiv -4 \equiv 0 \equiv 4 \equiv 8 \equiv 12 \equiv \pmod 4$

[1] $\ldots \equiv -7 \equiv -3 \equiv 1 \equiv 5 \equiv 9 \equiv 13 \equiv \pmod 4$

[2] $\ldots \equiv -6 \equiv -2 \equiv 2 \equiv 6 \equiv 10 \equiv 14 \equiv \pmod 4$

[3] $\ldots \equiv -5 \equiv -3 \equiv 3 \equiv 7 \equiv 11 \equiv 15 \equiv \pmod 4$

This way, the addition, subtraction or multiplication of congruences, reduces to that of their residue classes. For example, we can add two residue classes [a] and [b] by defining their sum to be the residue class containing a+b.

For instance [24] +[35] =[19] (mod 40) which can be verified by

```
Mod[24+35,40]
19.
```

Exercise Set 1.1

1. Is $8^{600} \equiv 6^{800} \pmod 7$?

2. How would you test to see if an integer m is divisible by another integer n, using the command **Mod**?

3. Use the command **Mod** to add [678] and [543] , modulo 700.

4. Find the residue classes of
   a) 6! modulo 7
   b) 10! modulo 11
   c) 12! modulo 13
   d) 16! modulo 17
   e) Can you propose a theorem from the above congruences?

5. Find the residue classes of
   a) 3^{10} modulo 11
   b) 5^{12} modulo 13
   c) 8^{22} modulo 23
   d) 39^{36} modulo 37
   e) Can you propose a theorem from the above congruences?

The answers to parts e) of problems 4 and 5 of exercise set 1.1 form the following two beautiful results: Wilson's Theorem and Fermat's Little Theorem. Apparently neither result was first proved by the person by whose name they are remembered - a not uncommon occurrence in mathematics. The proof of these theorems can be found in [Rosen].

Theorem 1.1. Wilson's Theorem: If p is prime, then $(p-1)! \equiv -1 \pmod{p}$.

The first proof of Wilson's Theorem was given by the French mathematician Joseph Lagrange on 1770. Wilson, conjectured, but did not prove it.

Theorem 1.2. Fermat's Little Theorem: Let x be any integer and p any prime not dividing x, then

$$x^{p-1} \equiv 1 \pmod{p}.$$

The *order* of an integer a modulo d, is the *minimal* value of *n* for which $a^n \equiv 1 \pmod{d}$. For example the order of 15 modular 22 is 5, since 5 is the minimal value for which $15^5 \equiv 1 \pmod{22}$. You can verify this using

```
PowerMod[15,5,22]
1
```

There are of course other integers *n* for which $15^n \equiv 1 \pmod{22}$. For example n=10, 15 or 20, but 5 is the minimal.

We can write a MATHEMATICA program for computing the order of a (mod d).

New MATHEMATICA Commands:

`While[test, body]` evaluates test, then body, repetitively, until test first fails to give True.

The symbol **!=** means not equal. **n++** means to increment n by 1.

Program 1.1: PrintOrder

```
**********************************************************
     PrintOrder[a_Integer,d_Integer?Positive]:=
     Module[{n=1},While[PowerMod[a,n,d] != 1,n++];
     Print[" The order of ",a," Mod ",d," is ",n]]

**********************************************************
```

```
PrintOrder[107,201]
The order of 107 Mod 201 is 22
```

Make sure, that *a* and *d* are coprime, otherwise there is no such order.
(Why? Try a=12, d=15)

Exercise Set 1.2

Experiment with MATHEMATICA to determine the following:

1. Is the converse of Wilson's theorem true? i.e. If *n* is a positive integer such that $(n-1)! \equiv -1 \pmod{n}$, then *n* is prime.

2. Is the converse of Fermat's little theorem also true? i.e. If *n* is a positive integer such that $x^{n-1} \equiv 1 \pmod{n}$, then *n* is a prime.

3. Modify the program **PrintOrder** to get a list of all integers *n* for which $a^n \equiv 1 \pmod{d}$.

4. Using the program **PrintOrder**, investigate, for a given prime *p*, the list of all possible orders of integers, *x*: obviously one need only compute the orders of the integers 1, 2, 3, ..., p–1. What do you notice about these orders? Can you propose a theorem from the above orders?

1.2. Modular Equations.

A modular equation is an equation of the form $a_n x^n + a_{n-1}x^{n-1}+\ldots+a_0 \equiv 0 \pmod d$ for which the coefficients are mod d, i.e., $0 \le a_i \le d-1$.

Example 1.4. $4x^3+5x^2+3x+2 \equiv 0 \pmod 7$.

A lot of times, we are interested in solving such modular equations for which MATHEMATICA has a built in command.

```
Solve[4x^3 +5x^2 + 3 x + 2 == 0 && Modulus == 7,x]
```

```
{{Modulus->7,x->-1},{Modulus->7,x->-3},{Modulus->7,x->-6}}.
```

The simplest Modular equation is a *linear* equation of the form $a x + b \equiv 0 \pmod d$, or it can be written as $a x \equiv b \pmod d$. The question is does such an equation always have a solution?

The answer is given by the following theorem:

Theorem 1.3. Let a, b, and d be integers with d>0 and (a,d)=m. If m does not divide b, then $a x \equiv b \pmod d$ has no solution. If $m \mid b$, then $a x \equiv b \pmod d$ has m solutions.

Example 1.5.

```
 sol = Solve[31 x + 22 == 0 && Modulus == 37,x]
{{Modulus -> 37, x -> 16}}
```

Since 37 is a prime number we have one solution, namely, x=16.

In order to obtain just the solution, without the extra brackets, we can use the following command:

```
sol[[1,2,2]]
```

```
16
```

To understand why we choose `[[1,2,2]]`, we must look at the `FullForm[sol]` which outputs the internal form of expression.

```
FullForm[sol]
List[List[Rule[Modulus, 37], Rule[x, 16]]]
```

Using `[[1,2,2]]`, we are reaching the second expression, namely, `16`, inside the second expression, namely, `Rule[x, 16]`, inside the first bracket, namely, `List[Rule[Modulus, 37], Rule[x, 16]]`.

We consider a few other examples:

Example 1.6. The case where the modulus d is a non-prime number.

```
sol = Solve[128 x == 833 && Modulus ==1001,x]
```

```
Roots::modp:Value of option Modulus->1001 should be a prime number.
 {ToRules[Modulus==1001 && Roots[x==-189,x,Modulus -> 1001]]}
```

As we can see, MATHEMATICA doesn't find the positive solution of a linear modular equation if the modulus is a non-prime. However, it does provide us with the solution x= –189 which is equivalent to x=1001–189 = 812. Note that in this case, there is only one solution since gcd(128,1001)=1.

In order to obtain the solution without the extra output, which is –189, we can use **`sol[[1,1,2,1,2]]`**. Why do we choose `[[1,1,2,1,2]]`?

Example 1.7. The case where the modulus *d* is a non-prime and gcd(a,d) does not divide *b*.

```
Solve[87 x ==61 && Modulus == 159,x].
```

```
{{Modulus -> 53, x -> -6}}
```

Since gcd (87,159)=3, but 3 does not divide 61, there is no solution. The output indicates that there is a solution of the equation mod 53, which is a prime factor of 159.

Example 1.8. The case where the modulus *d* is a non-prime number and gcd(a,d) | b.

```
Solve[52 x == 24 && Modulus ==76,x]
```

```
Roots::modp:Value of option Modulus->76 should be a prime number.
{ToRules[Modulus == 76 && Roots[x == 18, x, Modulus -> 76]]}
```

As gcd(52,76)=4, and 4 | 24, there are four solutions but since 76 is not a prime,

MATHEMATICA doesn't find all four solutions but finds the particular solution x=18, by first reducing the equation $52x \equiv 24 \pmod{76}$ by 4 and then obtaining the equation $13x \equiv 6 \pmod{19}$ and then solving this equation. we can find the other three solutions by adding multiples of 19, to the particular solution. Therefore, $x_0=18$, $x_1=19+18=37$, $x_2 = 2.19+18 = 57$ and $x_3 = 3.19+18 = 75$.

We can write a MATHEMATICA program to solve any linear modular equation and find all its solutions.

As we can see from the outputs obtained in the examples above, we need to consider three different cases: *d* is a prime number, *d* is not a prime but when it is reduced by the gcd(a,d), then it is a prime number and finally *d* is not a prime even after it is reduced by the gcd(a,d). Since we have two types of output forms: sol ={{Modulus -> d, x -> a}}, and sol= {ToRules[Modulus == d && Roots[x == a x, Modulus -> d]]}, we must use `s=sol[[1,2,2]], s=sol[[1,1,2,1,2]]` accordingly to obtain the right value corresponding to the solution in the output.

New MATHEMATICA Commands:

`GCD[a,d]` gives the greatest common divisor of the integers a and b.

`PrimeQ[d]` yield True if d is a prime number, and yields False otherwise.

`Table[f,{i,n}]` builds a length-n vector by evaluating f with i=1, i=2,..., i=n.

`If[condition, t, f]` gives t if condition evaluates to True, and f if it evaluates to False.

Remark: Before you run the program, use the Off command `Off[Roots::modp]` to stop printing the messages that appear when *d* is not a prime number. i.e. messages of the form "`Roots::modp: Value of option Modulus -> d should be a prime number`".

Program 1.2: ModularEquation

```
**********************************************************
      ModularEquation[a_,b_,d_]:=(
 (*Remark:  Give the solutions to a x == b (mod d)*)
      g=GCD[a,d];
      If[PrimeQ[d],
```

```
      sol=Solve[a x ==b && Modulus == d,x][[1,2,2]];
(*Remark:  Give the solution as a positive number*)
      If[sol<0,sol=d+sol];
      Print["There is one solution x = ",sol],
       d1=d/g;
      If[ Mod[b,g]==0 ,
      Print["There are  ", g, "  solutions"];
      sol=Solve[(a/g) x ==(b/g) && Modulus == d1,x];
      If[PrimeQ[d1],
      s=sol[[1,2,2]],
      s=sol[[1,1,2,1,2]]];
(*Remark:  Give the solutions as positive numbers*)
      If[s<0,
      Print[Table[s + i d1,{i,1,g}]],
      Print[Table[s + i d1,{i,0,g-1}]]],
      Print["No solution"]]])
**********************************************************************
```

We can use **ModularEquation** to find the solutions of the equation 52 x ≡ 24 (mod 76),

```
ModularEquation[52,24,76]
There are  4  solutions
{18, 37, 56, 75}.
```

Definition 1.2. Given an integer *a* with gcd (a,d)=1, a solution of a x ≡1(mod d) is called an *inverse of a modulo d.*

Example 1.9. Since the solution of 26 x ≡1 (mod 63) is x=17. Therefore 17 is the inverse of 26, modular 63.

One way of finding the inverse of a number *a* is by using the program **ModularEquation.**

```
ModularEquation[26,1,63]
There are  1  solutions
{17}.
```

An easier way of finding inverse of a number *a,* molulo *d* is however to use the built in command **PowerMod[a,-1,d]**. How does this work?

```
PowerMod[26,-1,63]
17
```

Exercise Set 1.3

1. Use the program **ModularEquation** to find the solutions of the following equations. Does Theorem 1.3 hold?
   a) 27 x ≡450(mod 3609).
   b) 271 x ≡4502(mod 36096).
   c) 1723 x ≡2051 (mod 3446).
   d) 2145 x ≡1992 (mod 2341).
   e) Try a few equations of your own.

2. An astronomer knows that a satellite orbits the earth in a period that is an exact multiple of 1 hour. If the astronomer notes that the satellite completes 11 orbits in 17 hours, how long is the orbital period of the satellite?

3. Monkey puzzle: Five sailors were cast away on an island. They collected all the coconuts they could find. During the night one of the sailors woke up and divided the nuts into five equal piles and discovered that one nut was left over. He threw the extra one to the monkeys; he then hid his share and went back to sleep. Later a second sailor woke up and did the same as the first. He divided the remaining nuts into five equal piles, discovered there was one left over which he threw to the monkeys, and hid his hare. The other three sailors did the same, each throwing a single coconut to the monkeys. The next morning the sailors, ignorant of each others' deception, divided the remaining nuts into five equal piles, no nuts being left over this time. Find the smallest number of coconuts in the original pile.

4. Find an inverse for the following numbers modulo the given numbers:
   a) 25(mod 144).
   b) 15 (mod 676).
   c) 223 (mod 4578).

5. Write a simple program using **PowerMod** to find all numbers which are their

own inverses modular the prime numbers 5, 11, 23, 31, 43, 59. (For example 1 and 16 are their own inverse modular 17). Can you propose a theorem from observing the above inverses?

1.3 An application of Modular arithmetic.

Day of the week. Since the days of the week form a cycle of length seven, we can use modular arithmetic (mod 7) to derive a formula that gives us the day of the week of any day of any Gregorian year. We denote each day of the week by a number from 0 to 6.

- Sunday = 0,
- Monday = 1,
- Tuesday = 2,
- Wednesday = 3,
- Thursday = 4,
- Friday = 5,
- Saturday = 6.

We must first make some adjustments. Because the extra day in a leap year comes at the end of February, for our calculations, we renumber the months, starting with March.

- March = 1,
- April = 2,
- May = 3,
- June = 4,
- July = 5,
- August = 6,
- September = 7,
- October = 8,
- November = 9,
- December = 10,
- January = 11,
- February = 12.

Note that with this renumbering, the months of January and February are part of the preceding year. For instance, February 1886, is considered the 12*th* month of 1985. With this convention. Let

- k = day of the month,
- m = month,

- c = century,
- y = year in the century,
- n = year.

For instance, for January 11, 1952, we have k=11, m=11, c=19, y =51.

To find W, the day of the week of the day *k* of month *m* of year *n*, we use the formula

$W \equiv k + [2.6\ m - 0.2] - 2\ c + y + [\frac{c}{4}] + [\frac{y}{4}] \pmod 7$. See [Rosen] for the details.

We can write a simple MATHEMATICA program to find the day of the week corresponding to any date in the Gregorian calendar.

New MATHEMATICA Command:

`Floor[x] gives the greatest integer less than or equal to x.`

Program 1.3: day

```
*************************************************************
```

*(*Remark: The dates must be entered according to the adjusted table given above.*)*

```
day[month_,day_,year_]:=( c=Floor[year/100];
                          y=year-100 c;
                          m=month;
                          k=day;
Print[Mod[k +Floor[2.6 m - 0.2]
 - 2 c +y +Floor[c/4] +Floor[y/4],7]])
*************************************************************
```

To find the day of the week corresponding to January 1, 2000, we must enter:

```
day[11,1,1999]
```

6

So, the first day of the twenty-first century is Saturday.

Note: The Gregorian calendar started on September 14, 1752. Before then, Julian calendar was used. For a more detailed discussion of calendars, see [Vardi].

Exercise Set 1.4

1. Modify the program **day**, so that you can input the dates as we know them without the adjustments. For instance **day[1,1,2000]** for the first day of the twenty-first century. Also modify the program so that the out put would

print the day. For instance it would print 'Saturday ' instead of '6'.

2. Find the day of the week of the day you, and any one close to you were born.

3. Find the day of the week of the following important dates in U. S. history.
   a) July 4, 1776 (US. Declaration of Independence)
   b) December 8, 1941 (Japanese attack on Pearl Harbor)
   c) January 8, (Martin Luther King's Birthday)
   d) November 22, 1963 (Assassination of J.F. Kennedy)
   e) July 20, 1969 (First man on the moon)

4. More recent calculations have shown that the true length of the year is approximately 365.2422 days. To correct this small discrepancy between the number of days in a year of the Gregorian calendar and an actual year, it has been suggested that the years exactly divisible by 4000 should not be leap years. Adjust the formula for the day of the week and modify the program day to take this correction into account. Test your program for some future dates.

5. The Julian Calendar: In the Julian calendar which started in 46 B.C., it was assumed that the length of the year was 365.25 instead of 365.2425 which is used in the Gregorian calendar. On September 3, 1752 by which time the Julian calendar was off by 11 days, Great Britain and America, the Gregorian calendar was adopted and September 3, 1752 in the Julian calendar became September 14, 1752 in the Gregorian calendar. Modify the program **`day`**, so that you can implement the Julian calendar. Test your program for
   October 12, 1492, the day Columbus discovered America which was a Friday.

1.4. Quadratic Modular Equations.

One very central topic in number theory is the study of quadratic modular equations of the form $x^2 \equiv a \pmod{p}$, where p is an odd prime. If this equation has a solution x we shall say that a is a quadratic residue of p (i.e. is the remainder left from some square after division by p). For example 2 is a quadratic residue of 7 since $3^2 \equiv 2 \pmod{7}$. If the equation has no solution we shall say that a is a quadratic non-residue of p. Taking p = 7 you can easily check that 1, 2 and 4 are quadratic residues while 3, 5, and 6 are quadratic non-residues.

We can write a MATHEMATICA program to determine if an integer is a quadratic residue of a prime number.

New MATHEMATICA Command:

`IntegerQ[expr] gives True if` *`expr`* `is an integer, and False otherwise.`

Program 1.4: residue

```
*****************************************************************
     residue[i_,p_] := If[IntegerQ[
          Solve[x^2 ==i && Modulus == p,x][[1,2,2]]],
               Print[i," is a residue"],
                    Print[i,"   is a non-residue  "]]
*****************************************************************
```

```
residue[25,29]
25 is a residue

residue[17,29]
17 is a non-residue
```

We can also apply the program **`residue`** as a function to any lists of numbers to see whether the elements of this list are residues or non-residues with respect to a given prime: For example for the prime number p=17,

```
residue[#,17]& /@ {3,7,11,17}

3   is a non-residue
7   is a non-residue
11   is a non-residue
17   is a residue
```

A one-liner using the program **`residue`** and the command **`Range[n]`** which generates the list {1, 2, ..., n}, can be formed to find all residues and non-residues of a prime number.

Program 1.5: Allresidues

```
*****************************************************************
```

```
Allresidues[p_] := (residue[#,p]& /@ Range[p-1];)
**********************************************************

Allresidues[7]

1    is a residue
2    is a residue
3    is a non-residue
4    is a residue
5    is a non-residue
6    is a non-residue
```

Exercise Set 1.5

Use the program **Allresidues** in the following exercises.

1. Check that there are always $\frac{1}{2}(p-1)$ residues and $\frac{1}{2}(p-1)$ non-residues for any odd prime p. (You can prove this by showing that $1^2, 2^2, 3^2,...,(\frac{1}{2}(p-1))^2$ are all incongruent modulo *p*, while $(p-r)^2$ and $r^2$are all incongruent modulo *p*, while $(p-r)^2$ and r^2 are congruent modulo p.)

2. Check that the product of two quadratic residues is a residue. How about the product of two non-residues?

3. Define the Legendre symbol $(\frac{a}{p})$ by

$$(\frac{a}{p}) = \begin{cases} +1, & \text{if } a \text{ is a quadratic residue mod } p, \\ -1, & \text{if } a \text{ is a quadratic non-residue mod } p. \end{cases}$$

Modify the program **residue** to find the Legendre symbol $(\frac{a}{p})$.

a) Can you find a relationship between $(\frac{a}{p})$, $(\frac{b}{p})$ and $(\frac{ab}{p})$.

b) Check that if p is an odd prime and a is not divisible by p then

$$(p-1)! \equiv -(\frac{a}{p})\, a^{(p-1)/2} \pmod{p}.$$

For example $6! \equiv 5^3 \pmod 7$. The proof of this result is not very difficult.

c) Deduce Wilson's Theorem from the result stated in b.

d) Check the validity of the following famous theorem:

Gauss's Law of Reciprocity. If p and q are odd primes then

$\left(\frac{p}{q}\right) = \left(\frac{q}{p}\right)$ *unless p and q are both of the form 4n+3, in which case* $\left(\frac{p}{q}\right) = -\left(\frac{q}{p}\right)$.

The proof of this result is not obvious.[Hardy and Wright]

The Law of Reciprocity links the solvability of the equation $x^2 \equiv p \pmod q$, with that of $x^2 \equiv q \pmod p$. Therefore by using this law we can simplify the computation of Legendre symbols.

For example:

$$\left(\frac{345}{431}\right) = \left(\frac{3}{431}\right)\left(\frac{5}{431}\right)\left(\frac{23}{431}\right)$$
$$= \left(\frac{431}{3}\right)\left(\frac{431}{5}\right)\left(\frac{431}{23}\right)$$

Since $\left(\frac{431}{3}\right)$ is equivalent to $\left(\frac{2}{3}\right)$, $\left(\frac{431}{5}\right)$ is equivalent to $\left(\frac{1}{5}\right)$ and $\left(\frac{431}{23}\right)$ is equivalent to $\left(\frac{17}{23}\right)$, we have

$$\left(\frac{345}{431}\right) = \left(\frac{2}{3}\right)\left(\frac{1}{5}\right)\left(\frac{17}{23}\right) = (-1)(+1)(-1) = +1.$$

For large primes, one can write a program which computes $\left(\frac{a}{p}\right)$ by this method.

1.5. Systems of simultaneous congruences.

Consider the following problem. Find a number that leaves a remainder of 1 when divided by 3, a remainder of 2 when divided by 5 and a remainder of 3 when divided by 7. This problem leads to the following system of congruences

$x \equiv 1 \pmod 3$, $x \equiv 2 \pmod 5$, $x \equiv 3 \pmod 7$.

These types of problems arose in the puzzles that were considered by the Chinese mathematicians as early as the first century. The following theorem which states that certain systems of linear congruences can always be solved derives its name from the ancient Chinese heritage of the problem.

Theorem 1.4. The Chinese Remainder Theorem. Let $m_1, m_2, \ldots, m_r$ be pairwise relatively prime positive integers. Then the system of congruence

$x \equiv a_1 \pmod{m_1}$,

$x \equiv a_2 \pmod{m_2}$,

.

.

.

$x \equiv a_r \pmod{m_r}$,

has a unique solution modulo $m = m_1 m_2 ... m_r$.

Proof. We can construct a solution to the system. Let $M_k = \frac{m}{m_k}$. Since m_i's are pairwise relatively prime, $M_k$and m_k are also relatively prime. We can now find an inverse y_k of $M_k$modulo m_k, i.e. $M_k\ y_k \equiv 1 \pmod{m_k}$. We now form

$$x \equiv a_1 M_1 y_1 + a_2 M_2 y_2 + ... + a_r M_r y_r \pmod{m}$$

We leave it as an exercise for the reader to show that *x* is a simultaneous solution of the *r* congruences and any other solution is congruent to *x*, modulo m.

To illustrate the use of the Chinese remainder theorem, we can solve the system

$x \equiv 1 \pmod 3$, $x \equiv 2 \pmod 5$, $x \equiv 3 \pmod 7$.

$m = 3.5.7 = 105$, $M_1 = 105/3 = 35$, $M_2 = 105/5 = 21$, $M_3 = 105/7 = 15$. Solving for the inverses, we get $y_1 \equiv 2 \pmod 3$, $y_2 \equiv 1 \pmod 5$, $y_3 \equiv 1 \pmod 7$.

Hence, $x \equiv 1.35.2 + 2.21.1 + 3.15.1 \pmod{105}$

$\equiv 52 \pmod{105}$.

We can use this constructive method to write a MATHEMATICA program that solves such systems of congruences.

New MATHEMATICA Commands:

If `list1 =`$\{a_1, a_2, . . ., a_n\}$ and `list2 =`$\{b_1, b_2, . . ., b_n\}$, then `Times @@ list1` multiplies the elements of the list, i.e. $a_1 . a_2 . . . a_n$.
`list1*list2` forms a new list $\{a_1b_1, a_2b_2, . . ., a_nb_n\}$.
`Apply[Plus,list1]` adds the elements of the list, i.e. $a_1 + a_2 + . . . + a_n$.
Also note that PowerMod is listable, i.e.`PowerMod[list1,-1,list2]` gives a new list namely,
{PowerMod[a_1,-1,b_1], PowerMod[a_2,-1,b_2], . . . , PowerMod[a_n,-1,b_n]}.

Program 1.6: ChineseRemainder

```
**************************************************************
ChineseRemainder[alist_,mlist_]:=(
      m = Times @@ mlist;
      mm=m/mlist;
x=Mod[Apply[Plus,alist*mm*PowerMod[mm,-1,mlist]],m];
      Print["The answer is ",x," Mod ", m])
**************************************************************
```

Here is the solution to the system

$x \equiv 1 \pmod 3$, $x \equiv 2 \pmod 5$, $x \equiv 3 \pmod 7$.

```
ChineseRemainder[{1,2,3},{3,5,7}]

  The answer is 52 Mod 105
```

We can check to see if we have the right answer. Snce **Mod** is listable, we only need

```
Mod[52,{3,5,7}]
```

{1, 2, 3}.

Exercise Set 1.6

Use the program **ChineseRemainder** in the following problems.

1. Find the least positive solution to the following system of linear congruences.
   $x \equiv 3 \pmod{17}$,
   $x \equiv 12 \pmod{23}$,
   $x \equiv 5 \pmod{21}$,
   $x \equiv 13 \pmod{26}$,
   $x \equiv 2 \pmod{53}$,
   $x \equiv 33 \pmod{55}$.

2. Find the smallest positive integer that has the remainders 20, 32, 43 and 50 when divided by 25, 36, 49, 67, respectively.

3. The three children in a family have three sticks that are 3 inches, 4 inches, and 5 inches long. When they measure the length of a table in their house, they each find that there is one inch left over. How long is the table?

4. The program **ChineseRemainder** only solves systems of linear congruences when the m_i's are pairwise relatively prime. What is your output when you input at least two m_i's that are not relatively prime? Modify the program to print a message " No solution", when at least two m_i's are not relatively prime. You can use the fact that two integers *m* and *n* are relatively prime if and only if gcd(m,n) = 1.

5. Use the program **ChineseRemainder** in a MATHEMATICA routine to find a solution to the system of congruences
   $x \equiv 0 \pmod{4}$,
   $x \equiv -1 \pmod{9}$,
   $x \equiv -2 \pmod{25}$,

.

.

.

$x \equiv -k + 1 \pmod{p_k^2}$,

where p_k is the k-th prime. This shows that there are arbitrarily long stings of integers each divisible by a perfect square. Try your routine with k=5. What are the 5 consecutive numbers that are each divisible by a perfect square?

(The following commands could be useful: `Range[imin, imax, di]` and `Prime[n]`, which gives the *n*th prime can be useful here.)

6. Use the program **`ChineseRemainder`** to find a solution to the system of congruences

$5x \equiv 11 \pmod{17}$,

$9x \equiv 17 \pmod{25}$,

$3x \equiv 19 \pmod{32}$,

$11x \equiv 6 \pmod{37}$.

1.6. More on Systems of simultaneous congruences.

In general the moduli in the system of linear congruences are not necessarily relatively prime. In this case we have the following theorem.

Theorem 1.5. The Generalized Chinese Remainder Theorem. The system of congurences

$x \equiv a_1 \pmod{m_1}$,

$x \equiv a_2 \pmod{m_2}$,

.

.

.

$x \equiv a_r \pmod{m_r}$,

has a solution if and only if $\gcd(m_i, m_j) \mid (a_i - a_j)$ for all pairs (i,j), where $1 \le i \le j \le r$. Furthermore, if a solution exists, it is unique modulo $\text{lcm}[m_1, m_2, ..., m_r]$.

Proof. The proof of this theorem is done by using mathematical induction on r. In the case of r = 2, we can write the first congruence as $x = a_1 + k\, m_1$ where k is an integer, and then insert this expression for *x* into the second congruence, which results in $k\, m_1 \equiv a_2 - a_1 \pmod{m_2}$. Now since $\gcd(m_2, m_1) \mid (a_2 - a_1)$, we can reduce the congruence to another congruence modulo $\frac{m_2}{\gcd(m_2, m_1)}$, and solve for *k*.

We use this argument to write a MATHEMATICA program to solve a system consisting of two linear congruences.

Remark. In the same way as in program **Modular Equation**, we must use Off[Roots::modp], to stop printing unnecessary messages resulting from the Solve command before we run the program.

Program 1.7: CrtTwoEquations

```
**********************************************************************
      CrtTwoEquations[row1_,row2_]:=
      If[row1=={0,0},{0,0},
      Block[{x},
            a1=row1[[1]]; a2=row2[[1]];
            m1=row1[[2]]; m2=row2[[2]]];
      g=GCD[m1,m2];
```

*(*Remark: If the condition $gcd(m_2, m_1) \mid (a_2 - a_1)$ holds find the solution., otherwise print that there is no solution.)*

```
      If[IntegerQ[(a2-a1)/g],
```

*(*Remark: Define a function pq in order to use an If statement with three options.)*

```
      pq[1] = 0; pq[n_] := PrimeQ[n];
      s = If[pq[m2/g],
      Solve[(m1/g) x ==(a2-a1)/g
            && Modulus == m2/g,x][[1,2,2]],
      Solve[(m1/g) x ==(a2-a1)/g
            && Modulus == m2/g,x][[1,1,2,1,2]],0];
```

*(*Remark: If the solution exits give the smallest positive solution and the modulus, otherwise print there is no solution and return {0,0} for the output.)*

```
      If [s<0, s=s + m2/g];
{a1 + s m1,LCM[m1,m2]}, Print["No solution"];{0,0}]]
**********************************************************************
```

We can find the solution of the system of congruences x ≡ 24 (mod 45), x ≡ 51 (mod 72):

```
      CrtTwoEquations[{24,45},{51,72}]
      {339, 360}
```

We can also see that for the system of congruences x ≡ 124 (mod 145), x ≡ 28

(mod 60), where the condition gcd(m_2, m_1) | (a_2 – a_1) does not hold, the program returns "no solution".

```
CrtTwoEquations[{124,145},{28,60}]
No solution
```

Now that we have a program which can solve a system consisting of two linear congruences, we can iterate it to solve systems with several linear congruences.

New MATHEMATICA Commands:

```
FoldList[f, x, {a,b,...}] gives {x,f[x, a],f[f[x,a], b],...}.
First[expr] gives the first element in expr.
Rest[expr] gives expr with the first element removed.
```

Program 1.8: CrtAllEquations

```
**************************************************************
      CrtAllEquations[l_List] :=

      FoldList[CrtTwoEquations,First[l],Rest[l]]
**************************************************************
```

Let us use this program to solve the system of linear congruences

$x \equiv 2 \pmod{14}$,

$x \equiv 16 \pmod{21}$,

$x \equiv 12 \pmod{16}$,

$x \equiv 5 \pmod{17}$,

$x \equiv 4 \pmod{38}$,

$x \equiv 23 \pmod{47}$,

```
CrtAllEquations[{{2,14},{16,21},{12,16},{5,17},{4,38},{23,47}
}]

{{2, 14}, {16, 42}, {268, 336}, {940, 5712}, {98044, 108528},
{2594188, 5100816}}.
```

Note that the output gives a solution for each step of iteration, i.e. {2,14} is the solution for the first linear congruence, {16,42} is the solution for the first two equations taken simultaneously and so on. One could modify the program slightly to print only the last output in a more appropriate way.

New MATHEMATICA Command:

```
Last[expr] gives the last element in expr.
```

Program 1.9: CrtSolution

```
*************************************************************
     CrtSolution[l_List] := Module[{x},
          x = Last[CrtAllEquations[l]];
     If[x!={0,0},Print["The solution is ", x[[1]],
          " Modulus ",x[[2]]]]]
*************************************************************
```

```
CrtSolution[{{2,14},{16,21},{12,16},{5,17},{4,38},{23,47}}]

The solution is 2594188 Modulus 5100816
```

If the system has no solution, the program just returns "No solution".

```
CrtSolution[{{11,14},{12,13},{3,14},{4,15},{5,16},{0,17}}]

     No solution
```

Exercise Set 1.7

Use the program **CrtAllEquations** or **CrtSolution** to solve the following problems.

1. Solve the following system of linear congruences if you can:
   a) $x \equiv 102 \pmod{114}$,
      $x \equiv 159 \pmod{213}$,
      $x \equiv 204 \pmod{276}$,
      $x \equiv 228 \pmod{312}$.
   b) $x \equiv 121 \pmod{145}$,
      $x \equiv 126 \pmod{135}$,
      $x \equiv 206 \pmod{265}$,
      $x \equiv 155 \pmod{315}$.

2. Solve the following ancient Indian problem: If eggs are removed from a basket two, three, four, five and six at a time, there remain, respectively, one, two, three, four and five eggs. But if the eggs are removed seven at a time no eggs remain. What is the least number of eggs that could have been in the basket?

3. A small boy, playing with a pile of blocks, notes that he has 5 blocks too few to

make a solid whose rectangular base contains 16 blocks. However, he has 1 block too many if he makes a solid whose rectangular base has 25 blocks. If the boy makes a solid whose rectangular base has 20 blocks, all the blocks are used. Determine the least number of blocks that the boy is playing with.

4. a) What condition is necessary for a system of the following form to have a solution?

$a_1 x \equiv b_1 \pmod{m_1}$,

$a_2 x \equiv b_2 \pmod{m_2}$,

:

:

$a_r x \equiv b_r \pmod{m_r}$.

b) Use the program **CrtAllEquations** to solve the systems

$3x \equiv 11 \pmod{14}$,

$7x \equiv 11 \pmod{15}$,

$6x \equiv 15 \pmod{21}$,

$5x \equiv 21 \pmod{32}$.

c) Modify the program **CrtAllEquations** to solve systems of the form given in part a).

Chapter 2. Diophantine Equations

2.1. Linear Diophantine Equations.

Consider the following problem. A woman wishes to mail a package. It is determined that the postage necessary to send this package first class is \$1.75 but only 29-cent and 15-cent stamps are available. Can some combination of these stamps be used to mail the package? If we let *x* denote the number of 29-cent stamps and *y* the number of 15-cent stamps, then the equation $29x + 15y = 175$ must be satisfied. To solve this problem, we need to find all solutions of this equation, where both *x* and *y* are non-negative integers. One such solution is x=5 and y = 2.

Definition 2.1. An equation in which all coefficients are integers and we are interested only in integer solutions is called a *Diophantine Equation*.

Diophantine equations are named after the ancient Greek mathematician *Diophantus* (c. 250 C.E.). Diophantine equations of the form $ax + by = c$, where *a*,*b*, and *c* are integers are called *linear diophantine equations* in two variables. In general, we can have other types of Diophantine equations such as $x^2 - y^2 = 1$, and $\frac{1}{x} + \frac{1}{y} = \frac{1}{2}$ which are Quadratic Diophantine Equations and $x^2+2 = y^3$ which is a Cubic Diophantine Equation.

As another example, we can consider the equation $x^n + y^n = z^n$, which arises in Fermat's Last Theorem which states that this equation for any integer *n* greater than two, has no positive integer solutions for *x*, *y*, and *z*. To read more on Fermat's Last Theorem, which is still an open problem, See [Edwards].

In general solving non linear Diophantine Equations can be extremely difficult.

Alinear Diophantine Equation could have infinitely many solutions, one solution or no solution. For example the linear equation $3x + 5y = 1$ has infinitely many solutions such as x=2, y =–1; or x=–3, y=2.

Theorem 2.1. Let $ax + by = c$ be a Diophantine Equation with d= GCD(a, b). This equation has no integer solutions if c is not divisible by d. If c is divisible by d, then there are infinitely many integers solutions. Moreover, if $x = x_0$, $y = y_0$ is a particular solution of the equation, then all solutions are given by

$$x = x_0 + \frac{b}{d}n, \quad y = y_0 - \frac{a}{d}n, \text{ where } n \text{ is an integer.}$$

Prove the theorem. Do you see any resemblance between this theorem and Theorem

1.3?

Example 2.1. 4x + 6y = 1 has no integer solutions since gcd(4,6) =2 which doesn't divide c.

The following program makes a table of *n* solutions to a Linear Diophantine equation if an integral solution exits.

New MATHEMATICA Command:

`ExtendedGCD[a,b] returns the list {d,{s,t}} where d=as +bt.`

Program 2.1: LinearDiophantine

```
**********************************************************************
      LinearDiophantine[a_,b_,c_,n_]:=
      (*  Remark  this gives n solutions to a x + b y == c *)
      ({d,{s,t}}=ExtendedGCD[a,b];
      x0=c s/d;
      y0=c t/d;
      If[Mod[c,d]==0,
      Print[
      Table[{x0,y0}+ {b/d i ,-a/d i}, {i,0,n-1}]],
      Print["No solution"]])
**********************************************************************
```

To obtain four solutions of the equation 21 x + 14 y = 35, we use n=4.

```
LinearDiophantine[21,14,35,4]
```

which gives us

`{{5, -5}, {7, -8}, {9, -11}, {11, -14}}.`

Note: Always test one of the solutions that you get in the output to make sure that they are right. If a program is correct but produces wrong answers the second time you apply it, it might be that one of the local variables has taken on a value you don't want. In this case, you must either clear all the variables before using it again or use a Block or Module command for the local variables. If you want your program to be more elaborate create a package for the program.

Exercise Set 2.1

1. Solve the stamp problem using **LinearDiophantine**.

Use the package **LinearDiophantine**,

2. Determine 5 solutions for the following linear Diophantine equations:

   a) 69 x + 54 y = 387.

   b) 158 x + 806 y = 338.

3. Find a linear Diophantine equation that has no solution.

4. Can you find a linear Diophantine equation that has only positive solutions and one that has only negative solutions.

5. Modify the program **LinearDiophantine**, to print out only the positive solutions.

2.2. Continued Fractions.

One method used in solving Diophantine equations is the use of continued fractions.

Continued Fractions: A real number can always be decomposed into a single, infinitely continuing fraction of the form $a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ldots}}}$. Let us use π as an example. We know that $\pi = 3.14.....$ which tells us that $a_0 = 3$. In order to find a_1, we use

```
Solve[3 + 1/x  == Pi ,x ]
```

```
   1
{{x -> -(------)}}
        3 - Pi
```

```
%//N
```

```
{{x -> 7.06251}}
```

This indicated that $a_0 = 7$. Repeating the process

```
Solve[7 + 1/x  == 7.06251 ,x ]
```

```
{{x -> 15.9974}}
```

which gives us $a_2 = 15$, continuing the process we get $a_3 = 1$ and $a_4 = 292$ and so on.

Therefore $\pi = 3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \cfrac{1}{292 + \ldots}}}}$. A simpler notation for this continued fraction is [3,7,15,1,292,....].

The continued fraction expansion of rational numbers terminates after a finite number of terms. This fact can be seen applying the Euclid's Algorithm to the numerator and the denominator of the rational number.

Example 2.2. Consider $\frac{67}{28}$. Applying the Euclid's Algorithm to this number, we get $\frac{67}{28} = 2 + \frac{11}{28}$

$$\frac{28}{11} = 2 + \frac{6}{11}$$

$$\frac{11}{6} = 1 + \frac{5}{6}$$

$$\frac{6}{5} = 1 + \frac{1}{5}$$

And therefore $\frac{67}{28} = 2 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{5}}}$ or $\frac{67}{28} = [2,2,1,1,5]$.

The following program finds the desired number of terms in a continued fractions of any real number.

New MATHEMATICA Commands:

```
NestList[f, expr, n] gives a list of the results of applying
   f to expr 0 through n times.

Select[list, crit] picks out all elements ei of list for
   which crit[ei] is True.
```

For example we can select all even integers less than 20 by the following:

```
Select[Range[20],EvenQ]

{2, 4, 6, 8, 10, 12, 14, 16, 18, 20}
```

Program 2.2: ContinuedFraction

```
**********************************************************
     fraction[x_] := If[x==Floor[x],0,1/(x-Floor[x])];
ContinuedFraction[x_,n_] :=
       Select[Floor[NestList[fraction,x,n-1]],(#!=0)&]
**********************************************************
```

Find the first fifteen terms in the continued fraction of π.

Note that we must use **N[Pi]** to convert π to a real number.

```
ContinuedFraction[N[Pi],15]
{3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1}
```

Note that the program is modified to add 0 to the list, if x==Floor[x], i.e. if the algorithm terminates, which is the case if the number is a rational number. We then give only the non-zero terms in the list, using `(#!=0)&` in the command `Select`.

```
ContinuedFraction[67/28,15]
{2, 2, 1, 1, 5}
```

As we can see there are only five terms in the continued fraction expansion of $\frac{67}{28}$.

The following program puts the continued fraction expansion in a continued fraction form.

New MATHEMATICA Commands:

The following command are used for formatting:

```
Infix[f[e1, e2, ...]] prints with f[e1, e2, ...] given in
     default infix form: e1 ~ f ~ e2 ~ f ~ e3 ....
```

Example: **Infix[f[1,2,3],"+"]**

```
1+2+3
```

```
PrecedenceForm[expr, prec] prints with expr parenthesized as
      it would be if it contained an operator with precedenceprec.
```

PrecedenceForm acts as a "wrapper", which affects printing, but not evaluation.

Example:

```
a + PrecedenceForm[1/b,10]
```

```
a + (1/b)
```

In order to eliminate the parentheses from the output one must use a high number for the precedence.

```
a + PrecedenceForm[1/b,500]
```

```
a + 1/b
```

Program 2.3: ContinuedFractionForm

```
**************************************************************
FractionForm[a_,b_] := pf[b,1/a];
Format[pf[a_,b_]] :=
Infix[pf[a,PrecedenceForm[b,500]]," + "];
    (* Remark this puts the continued fraction expansion in a
       continued fraction form*)
    ContinuedFractionForm[l_List] :=
        Fold[FractionForm,Last[l],Rest[Reverse[l]]]
**************************************************************
ContinuedFractionForm[{2,2,1,1,5}]
```

```
                1
2 + ---------------------
                    1
    2 + -----------------
                      1
        1 + -------------
                        1
            1 + ---------
                        1
                    1 + -
                        5
```

One could write a program to reverse the procedure and find the real number corresponding to a given continued fraction expansion.

New MATHEMATICA Commands:

`Last[expr]` gives the last element in expr.

`Reverse[expr]` reverses the order of the elements in expr.

Program 2.4: AddContinuedFraction

```
*************************************************************
          AddFraction[a_,b_] := b + 1/a;
     AddContinuedFraction[l_List] :=
          Fold[AddFraction,Last[l],Rest[Reverse[l]]]
*************************************************************
AddContinuedFraction[{2,2,1,1,5}]

     67
     --
     28
```

This is in fact an illustration of the following theorem:

Theorem 2.2. Every finite continued fraction represents a rational number.

The proof of this theorem is not difficult and is left as an excercise.

Exercise Set 2.2

1. Find the first five terms of the continued fractions for $\sqrt{2}$ by hand.

2. Use **ContinuedFractionForm** to display the first 15 terms in the continued fraction of $\sqrt{2}$, in the continued fraction form.

3. Here is a rather tricky problem with a trivial solution.

   The problem is: What is the value of $1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cdots}}}}}$?

   Hint: This is the solution of the equation $1+\frac{1}{x} = x$ which leads to a quadratic equation. Why?

4. Find the first 20 terms of the continued fractions for the following real numbers, using the program **ContinuedFraction** and then add the results using **AddContinuedFraction**. Do you get back the original number?

   a) □

b) E

c) 13.4567

d) 2.7182835

5. Find the continued fraction for the following rational numbers:

a) $\frac{67}{28}$

b) $\frac{100}{29}$

c) $\frac{34}{21}$

6. Determine the rational number represented by each of the following continued fractions.

b) [0, 4, 5, 6, 2, 1]

a) [4, 7, 3, 5, 2, 1, 2, 11, 6]

c) [–13, 57, 28, 19, 33, 15]

7. Find the first 20 terms of the continued fractions for the following real numbers and explain what you notice:

a) $\sqrt{3}$

b) $1+\sqrt{3}$

c) $\sqrt{17}$

d) $1+\sqrt{17}$

e) $\frac{1+\sqrt{31}}{2}$

f) Try some other real numbers of the form $\frac{p+q\sqrt{m}}{r}$

Does the following Conjecture make sense:

Conjecture: A real number *c* has an infinite continued fraction expansion which eventually repeats if and only if *c* is of the form $\frac{p+q\sqrt{m}}{r}$ for some integers *p, q, r, m* with $m \geq 2$ a non-square. Such a number is called a quadratic irrationality. This conjecture is referred to as Lagrange's Theorem. The proof can be found in [Hardy and Wright].

8. Verify this with some more specific examples.

Using the idea of problem 8, we can find the value corresponding to any given infinite continued fraction expansion which eventually repeats.

Example 2.3. Find the value corresponding to the continued fraction expansion [5,4,8,4,8,4,8,.....] which we denote by $[5, \overline{4, 8}]$

We have $5 + \cfrac{1}{4 + \cfrac{1}{8 + \cfrac{1}{4 + \cfrac{1}{8 + \ldots}}}}$. If we consider $\cfrac{1}{4 + \cfrac{1}{8 + \cfrac{1}{4 + \cfrac{1}{8 + \ldots}}}}$ and let $x = \cfrac{1}{8 + \cfrac{1}{4 + \cfrac{1}{8 + \ldots}}}$, then we have $x = \cfrac{1}{4 + \cfrac{1}{8 + x}}$ which we can write as $x = \frac{8+x}{33+4x}$

and hence $4x^2 + 33x = 8 + x$ which simplifies to $x^2 + 8x - 2 = 0$

Taking the positive root of x, $x = -4 + 3\sqrt{2}$, we find that $[5, \overline{4, 8}] = 1 + 3\sqrt{2}$.

9. Modify the program **`AddContinuedFraction`** so that you can get the value corresponding to a repeating continued fraction expansion. Try your program on $[2, \overline{1, 1, 1, 4}]$.

2.3. Convergents of continued fractions.

Given a continued fraction $[a_0, a_1, a_2, ...]$, the convergents are simply the rational numbers obtained by truncating the continued fraction. Thus the convergents are given by the sequence
$\{c_1 = a_0, \; c_2 = [a_0, a_1], c_3 = [a_0, a_1, a_2], ...\}$.

For example the convergents of $\frac{100}{29}$ are given by the sequence $\{3, \frac{7}{2}, \frac{31}{9}, \frac{100}{29}\}$, and the convergents of $\sqrt{2}$ are given by the sequence $\{1, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \frac{41}{29}, ...\}$ which can be written as {1, 1.5, 1.4, 1.41667, 1.41379,.....}.

Note that the sequence of the convergents {1, 1.5, 1.4, 1.41667, 1.41379,.....} does indeed converge to the number $\sqrt{2}$.

Using the program **`AddContinuedFraction`**, we can write a simple program to find the sequence of convergents for a given continued fraction expansion.

New MATHEMATICA Commands:

```
Take[list, n] gives the first n elements of list.
```

For example:

```
Take[{1,2,3,4,5},#]& /@ Range[5]
{{1}, {1, 2}, {1, 2, 3}, {1, 2, 3, 4}, {1, 2, 3, 4, 5}}
Length[expr] gives the number of elements in expr.
```

Program 2.5: Convergents

```
*************************************************************
Convergents[l_List] := AddContinuedFraction /@
                       (Take[l,#]& /@ Range[Length[l]]
*************************************************************
```

For example we can see

```
Convergents[{1,20,3,1,2}]
```

$$\left\{1, \frac{21}{20}, \frac{64}{61}, \frac{85}{81}, \frac{234}{223}\right\}$$

Although each successive convergent of a continued fraction represents a better approximation to the value of the continued fraction, it is not efficient to compute each convergent from its definition, i.e. $c_n = [a_0, a_1, a_2,...,a_n]$. There are some recursive relations that enable us to compute each convergent from the preceding convergents and the terms of the continued fraction.

Theorem 2.3. If $c_n = \frac{p_n}{q_n}$, where c_n is the n-th convergent of the continued fraction $[a_0, a_1, a_2,...]$, then

$p_1 = a_1$, $p_2 = a_2a_1 + 1$,

$q_1 = 1$, $q_2 = a_2$,

and

$p_n = a_n p_{n-1} + p_{n-2}$ for $n \geq 3$,

$q_n = a_n q_{n-1} + q_{n-2}$ for $n \geq 3$.

The proof of this theorem is done by mathematical induction on $n \geq 3$. The reader can check the proof for n=3. If the recursive relations are true for all integers from 3 through k. Then

$c_k = [a_0, a_1, a_2,...,a_k] = \frac{p_k}{q_k}$, where $p_k = a_k p_{k-1} + p_{k-2}$ and $q_k = a_k q_{k-1} + q_{k-2}$. Now,

$c_{k+1} = [a_0, a_1, a_2, ..., a_k, a_{k+1}] = [a_0, a_1, a_2, ..., a_k + \frac{1}{a_{k+1}}]$. Therefore,

$$c_{k+1} = \frac{(a_k + \frac{1}{a_{k+1}})p_{k-1} + p_{k-2}}{(a_k + \frac{1}{a_{k+1}})q_{k-1} + q_{k-2}}$$

$$= \frac{a_{n+1}(a_k p_{k-1} + p_{k-2}) + p_{k-1}}{a_{n+1}(a_k q_{k-1} + q_{k-2}) + q_{k-1}}$$

$$= \frac{a_{k+1}p_k + p_{k-1}}{a_{k+1}q_k + q_{k-1}} = \frac{p_{k+1}}{q_{k+1}} .$$

The following program uses the algorithm described in theorem 2.3 to evaluate a sequence of the convergents of continued fraction expansion.

One way of applying a fuction *f* to a list is given by the following:

```
f/@ list
```

For example:

```
f[x_]:=x^2; f/@ {1,2,3,4}
```

{1, 4, 9, 16}

Program 2.6: ConvergentsDirect

```
**************************************************************
     DirectConvergents[a_List]:= Module[{p,q,c,n},
          p[0]=1; q[0]=p[-1]=0; q[-1]=1;
               p[n_]:=p[n]= a[[n]] p[n-1] + p[n-2];
               q[n_]:=q[n]= a[[n]] q[n-1] + q[n-2];
               c[n_]:= p[n]/q[n];
          c/@ Range[Length[a]]]
**************************************************************

     ConvergentsDirect[{1,3,5,1,2,4}]
         4   21   25   71   309
{1, -, --, --, --, ---}
         3   16   19   54   235
```

Using the idea given in the theorem we can also write a program to evaluate any

sequence of a desired length of the convergents of any real number .

New MATHEMATICA Commands:

`Do[expr, {imax}]` evaluates expr imax times.

`AppendTo[s, elem]` appends elem to the value of s, and resets s to the result.

`Divide @@ {x,y}` produces $\frac{x}{y}$.

`Break[ ]` exits the nearest enclosing Do, For or While.

Program 2.7: ConvergentsReal

```
**************************************************************
ConvergentsReal[x_, n_]:=
     Module[{p={0,1},q={1,0},xp=x,con={}},
          Do[
               {p,q} = {q,Floor[xp] q + p};
          AppendTo[con,Divide @@ q];
          If[xp == Floor[xp], Break[],
               xp=1/(xp-Floor[xp])],{n}]; con]
**************************************************************
```

```
ConvergentsReal[N[Sqrt[2]],10]

     3  7  17  41  99  239  577  1393  3363
{1, -, -, --, --, --, ---, ---, ----, ----}
     2  5  12  29  70  169  408  985   2378

N[%]
{1.,1.5,1.4,1.41667,1.41379,1.41429,1.4142,1.41422,1.4141,1.41421}
```

Exercise Set 2.3.

1. Find the convergents of each of the following continued fractions by using the program **ConvergentsDirect**.
   a) [14, 13 , 27, 6 , 4, 8, 12, 14]
   b) [1, 1, 1, 1, 1, 1, 1, 1, 1]
      Do you see any patterns emerging in the convergents of this continued fraction? Would you be able to guess what the convergents would be for [1, 1, 1,.....]?

c) [1, 2, 3, 4, 5]

Does the following relationship hold $p_5 = 4p_4 + 4p_3 + 3p_2 + 2p_1 + (p_1 + 1)$?

In general prove that for [1, 2, 3, 4, ..., n] we have,

$p_n = (n-1)p_{n-1} + (n-1)p_{n-2} + (n-2)p_{n-3} + \ldots + 3p_2 + 2p_1 + (p_1 + 1)$.

2. Modify the program **`ConvergentsDirect`** to find the *k*th convergent c_k, for a given continued fraction $[a_0, a_1, a_2, \ldots]$. Determine the 7th convergent of

the infinite continued fraction $[6, \overline{3, 12}]$.

3. Find the convergents for the following numbers using the program **`ConvergentsReal`** until they repeat i.e. $c_n = c_{n+1}$. what is the value of *n* in each case?

a) □

b) E

c) $\sqrt{31}$

4. Compare the convergents of a real number x and its reciprocal. Do you observe the following assertion:

The *n*th convergent of $\frac{1}{x}$ is equal to the reciprocal of the *(n-1)*th convergent of *x*. Can you prove it?

5. Combining the programs **`ContinuedFraction`** and **`Convergents`**, write a program to find the sequence of convergents for any real numbers. Test your program for a number of rational and irrational numbers. Compare the Timing of this program with the program **`ConvergentsReal`**.

6. Check and see if the following assertions are true:

a) The even convergents form a decreasing sequence,

$c_2 > c_4 > c_6 > \ldots$

b) The odd convergents form an increasing sequence,

$c_1 < c_3 < c_5 < \ldots$

c) Every odd convergent of an irrational number *x* is less than *x*, and every even convergent is greater than x,

$c_1 < c_3 < c_5 < \ldots < x < \ldots < c_6 < c_4 < c_2$.

7. Verify the following theorem for some finite continued fractions.

Theorem 2.4. If $c_n = \frac{p_n}{q_n}$, where c_n is the n-th convergent of the continued fraction $[a_0, a_1, a_2, ...]$, then

a) $p_n q_{n-1} - p_{n-1} q_n = (-1)^n$; that is, $c_n - c_{n-1} = \frac{(-1)^n}{q_n q_{n-1}}$,

b) $p_n q_{n-2} - p_{n-2} q_n = (-1)^n a_n$; that is, $c_n - c_{n-2} = \frac{(-1)^n a_n}{q_n q_{n-2}}$.

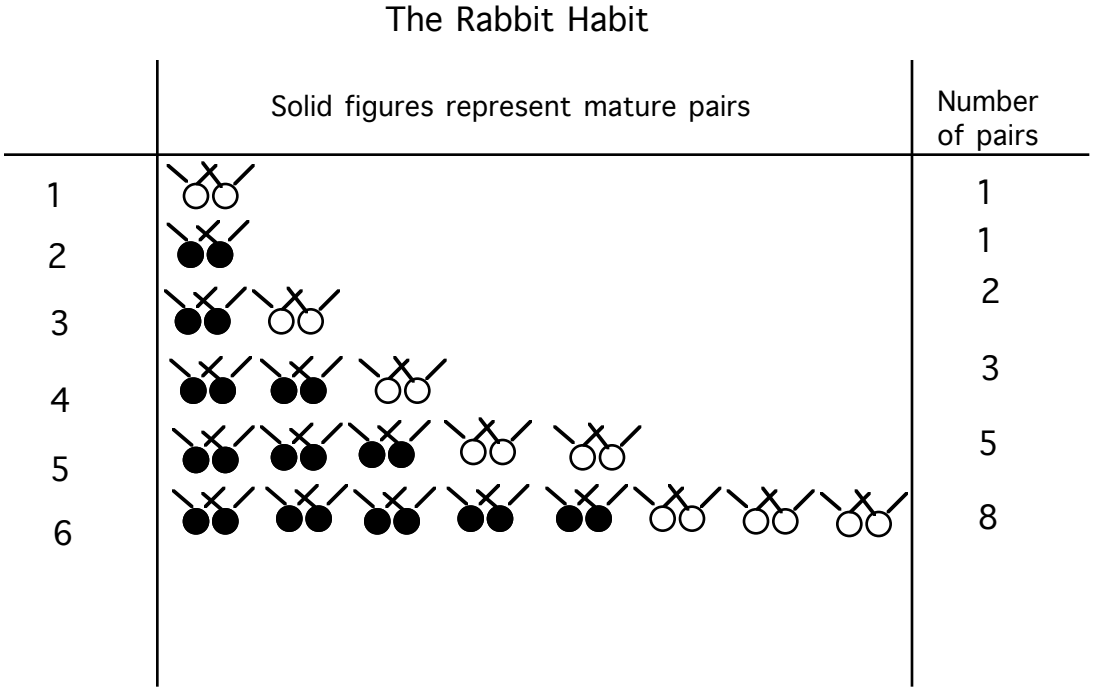
2.4. Continued Fractions and Fibonacci Numbers.

You have probably already come across the so called Fibonacci sequence

1, 1, 2, 3, 5, 8, 13, 21,.....

This sequence of numbers obtained the recursion relation $f_1=f_2= 1$, $f_n= f_{n-1}+f_{n-2}$ for $n\geq 3$ was found by Leonardo Fibonacci in 1202 as an answer to the following problem:

A pair of rabbits is mature enough to reproduce another pair after 2 months and will do so every month thereafter. If each new pair of rabbits has the same reproductive habits as its parents and none of them dies, how many pairs will there be at the end of n months? The following diagram show their reproductive habit.



Fibonacci, alias Filius Bonaccio or Leonardo di Pisa (ca. 1180-1250), was self-educated and came from the merchant class. During his extensive travels to Europe he learned Hindu and Arabic arithmetic and introduced Arabic numerals to Europe in his arithmetic book "liber abacci" in 1202, which was revised in 1228. This provided the premises for the further development of algebra. He was a member of the circle of scholars round Emperor Frederich II (1212-1250).

We can write a simple program in MATHEMATICA to calculate the values of the Fibonacci sequence:

Program 2.8: Fibonacci

```
*************************************************************
      Fibonacci[1]=Fibonacci[2]=1;

      Fibonacci[x_]:=Fibonacci[x]=
                       Fibonacci[x-1] + Fibonacci[x-2]
*************************************************************
```

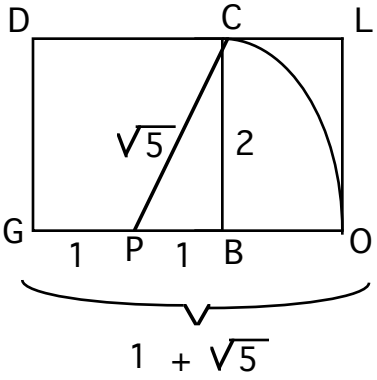
The Golden Ratio.

The Fibonacci Sequence has intrigued mathematicians for centuries, partly because it arises in unexpected places in nature and geometry. For example, in the heads of certain flowers, especially the sunflower, the seeds are distributed over the head in spirals, one set unwinding clockwise and the other counterclockwise. If one counts the numbers of these two kinds of spirals, they are almost always Fibonacci numbers. Small heads may have 13:21 or 21:34 combinations. Large heads 34:55, 55:89, or even 89:144 combinations.

one of the most remarkable properties of the Fibonacci sequence is that the ratio of its two consecutive terms (convergents of problem 3, exercise set 2.3) is alternately larger or smaller than the number $\phi = \frac{1+\sqrt{5}}{2} = 1.618033989....$

ϕ is referred to as the *Golden Ratio* or *Golden mean* and it takes its name from its progenitor, the golden rectangle, which is constructed as follows. Beginning with a square GBCD of side length 2, locate the midpoint P of one of the sides GB. Use P as center and PC which has length $\sqrt{5}$ as its radius to draw a circle determining a point on the extension of GB. Finally, locate L so that GOLD forms a rectangle, a so-called golden rectangle in which the ratio of the length to the width is the aforementioned golden ratio c. The Golden rectangle enchanted the Greeks and appears over and over in ancient architecture and can be found, for example in the Parthenon at Athens. For more applications of the Golden Ratio see [Coxeter] and [Hargittai].

The Golden Rectangle



Perhaps the most profound property of the golden mean is that, in a sense, it is the "most irrational" number in the number system, There are of course infinitely many irrational numbers such as π or e or $\sqrt{2}$. Each irrational number can be approximated arbitrarily close by ration numbers. However the expansion of an irrational number as a continued fraction yields the best rational approximation to the irrational, i.e., there exist no closer approximations to the irrational with denominators greater than the convergents of the continued fraction.

Theorem 2.5. If $\alpha = [a_0, a_1, a_2, ...]$ is an irrational number, then

$$\frac{1}{q_k^2(a_{k+1}+2)} \leq \left|\alpha - \frac{p_k}{q_k}\right| \leq \frac{1}{q_k^2 a_{k+1}}.$$

The proof of this theorem can be found in [Khinchin].

As the result of this inequality, it is clear that the approximations of irrational numbers with large values of a_k admit good approximations. Now the program **`ContinuedFraction`** applied to the golden mean $\phi = \frac{1+\sqrt{5}}{2}$ gives us the expansion [1, 1, 1, ...]. Therefore the approximations of ϕ are the worst approximations of any irrational number, since all values of a_k equal to 1. It is for this reason that ϕ is considered to be the "most irrational" number.

Exercise Set 2.4.

1. Find the first 90 terms of Fibonacci Sequence by using **`Fibonacci`**.

2. Check the following fact for different values of n, using the program **`Fibonacci`**. If f_n denotes the *n*th term of the Fibonacci sequence, then $f_{n+1} f_{n-1} - f_n^2 = (-1)^n$.

Try to prove this identity.

3. Use the program **ConvergentsReal** to find the convergents of the real number $\alpha = \frac{1+\sqrt{5}}{2}$. Examine each of the convergents $c_n = \frac{p(n)}{q(n)}$. What do you observe about p(n) and q(n)?

4. In 1724, the Swiss mathematician Daniel Bernoulli showed that the *n*th term of the Fibonacci sequence can be obtained by

$$f_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right].$$

Verify this assertion using MATHEMATICA and give a detailed proof by induction on n.

5. Refer to exercise 3, does c look familiar? What is the continued fraction expansion of c?

6. Use the program **ConvergentsDirect** to find an estimation for the number that corresponds to the continued fraction expansion [1,1,2,3,5,8,13,21,34,55,89]. Try with a continued fraction expansion with 30 Fibonacci numbers. Do you think the infinite continued fraction consisting of Fibonocci numbers converges? If so, to what number?

7. Write a MATHEMATICA program for the recursion formula
$k_1 = 1$, $k_n = 1 + \frac{1}{k_n}$ and find its first twenty terms. Do you observe anything?

8. *Open Problem:* Are there infinitely many Fibonacci numbers that are prime? This is the most famous unsolved problem involving the Fibonacci sequence.

2.5. Using Continued Fractions to solve linear Diophantine equations.

The theory of continued fractions can be used to derive solutions of the linear Diophantine equation ax + by = c, where a and b are relatively prime, for simplicity, we use the standard notation (a,b) = 1. If (a,b) =1, then the last convergent $c_n = \frac{p_n}{q_n}$ in the continued fraction expansion of $\frac{a}{b}$ is equal to $\frac{a}{b}$. Therefore, $p_n = a$ and $q_n = b$ and by the problem 7,

excercise set 2.3, $p_nq_{n-1} - p_{n-1}q_n = (-1)^n$. Therefore, $aq_{n-1} - p_{n-1}b = (-1)^n$. Multiplying both sides of this equation by $(-1)^n c$, we obtain

$$a[(-1)^n cq_{n-1}] + b\,[(-1)^{n+1} cp_{n-1}] = c.$$

Hence, a particular solution x_0 and y_0 of the linear Diophantine equation $ax + by = c$, is given by the equations

$$x_0 = (-1)^n cq_{n-1} \quad \text{and} \quad y_0 = (-1)^{n+1} cp_{n-1}.$$

Once we have the particular solutions, we can find other solutions by the equations
$x = x_0 + bt$ and $y = y_0 - at$, where t is an integer.

Note that in the equation $ax + by = c$, if *a* and *b* are not relatively prime but gcd(a,b) divides c, we can reduce the equation to one satisfying the condition (a,b) =1.

The following program which is a modification of the program **ConvergentsReal**, solves a linear Diophantine equation for a particular solution.

Program 2.9: Diophantine

```
**************************************************************
      Diophantine[a_,b_,c_]:=
      (*  Remark  this gives n solutions to a x + b y == c , using
            p_n and q_n  of the convergent c_n*)
      Module[{p={0,1},q={1,0},xp=a/b,cp=c/GCD[a,b],n=1,
            While[xp != Floor[xp],
                  {p,q} = {q,Floor[xp] q + p};
                        n=n+1;
                        xp=1/(xp-Floor[xp])];
            x0 = (-1)^n*cp*q[[2]];
            y0 = (-1)^(n + 1)*cp*q[[1]];
      (*  Remark  If b is negative the solution should be adjusted*)
      If[b<0, x0=-x0;y0=-y0];
            Print["x0 =",x0,"   y0 =",y0]]
**************************************************************
```

Determine a particular solution of the linear Diophantine equation
69 x + 54 y = 387.

```
Diophantine[69,54,387]

x0 =-903   y0 =1161
```

2.6. Solving Non-Linear Diophantine Equations. Pell's Method.

As our final use of continued fractions we show how they can be used to produce solutions to the Diophantine equation $x^2 - dy^2 = 1$ where *d* is an integer which is not itself a perfect square, e.g. $x^2 - 7\,y^2 = 1$ or $x^2 - 313\,y^2 = 1$. This equation , named after Pell always has infinite number of integer solutions.

We have seen by Lagrange's Theorem, exercise set 2.2, that the continued fraction expansion for $\sqrt{d}$ is eventually periodic of the form

$$[r(1), r(2),...,r(n), \overline{2r(1),r(2),...,r(n)}].$$

For example we can verify this for $\sqrt{7}$ and $\sqrt{313}$ using our program **ContinuedFraction**.

```
ContinuedFraction[N[Sqrt[7]],13]
```

```
{2, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4}
```

```
ContinuedFraction[N[Sqrt[31]],20]
```

```
{5,1,1,3,5,3,1,1,10,1,1,3,5,3,1,1,10,1,1,3}
```

Pell's Method. If (x,y) is an integral solution of the Diophantine equation $x^2 - d\,y^2 = 1$, where $d > 0$ is not a perfect square, then $\frac{x}{y}$ is a convergent of the simple continued fraction of $\sqrt{d}$.

The proof of this method can be found in [Hardy and Wright].

Pell's method indicates that for some convergent, $\frac{p(n)}{q(n)}$ of the continued fraction of $\sqrt{d}$, x=p(n) and y=q(n), would be a solution. For example for the equation $x^2 - 7\,y^2 = 1$, $p_3 = 8$ and $q_3 = 3$, would be a solution; checking: $8^2 - 7 \cdot 3^2 = 1$. The question is which *n*'s lead to a solution?

For a detailed discussion of this method see [Davenport], pp. 107-11.

The following program checks to see if x=p(n) and y=q(n) are solutions for the Diophantine equation $x^2 - d\,y^2 = 1$.

Program 2.10: Pell

```
*************************************************************
      (* Remark this finds the solutions to x^2–dy^2== 1, in the range [1,n]*)
      Pell[d_,n_]:=
            Module[{x,y,p={0,1},q={1,0},xp=N[Sqrt[d]]},
            g[x_,y_] := x^2 -d y^2;
Do[
      {p,q} = {q,Floor[xp] q + p};
            gq = g[q[[1]],q[[2]]];
If[gq ==1,
            Print["For n =",j," ",
      "{", q[[1]],",",q[[2]],"}","   is a solution."]];
            xp=1/(xp-Floor[xp]),{j,n}]]
*************************************************************
```

Find the solutions of the Diophantine equation $x^2 - 7y^2 = 1$.

```
Pell[7,30]
```

```
For n =4  {8,3}   is a solution.
For n =8       {127,48}   is a solution.
For n =12 {2024,765}   is a solution.
For n =16 {32257,12192}   is a solution.
For n =20 {514088,194307}   is a solution.
For n =24 {8193151,3096720}   is a solution.
For n =28       {130576328,49353213}   is a solution.
```

It is interesting to note that in this example for n=4k, k=1,2,3,...., there is always a solution.

Caution: Eventually computer error enters the calculations and we don't get any of the further solutions.

For example consider the Diophantine equation $x^2 - 13\ y^2 = 1$. Program **pell**, shows

```
Pell[13,100]
```

```
For  n  =10  {649,180}   is a solution.
```

```
For n =20 {842401,233640}   is a solution.
For n =30 {1093435849,303264540}   is a solution.
```

MATHEMATICA fails to compute the further solutions which we are guaranteed to have by the equation $x+y\sqrt{d} = (x_1+ y_1\sqrt{d})^r$, for $r \geq 1$, (Problem 4, excercise set 2.6) but a simple calculations shows that the result indeed holds.

```
Expand[(1093435849+303264540 Sqrt[13])^2]

2391203911756701601 + 663200639532988920*13^(1/2)

N[(2391203911756701601)^2-13 (663200639532988920 )^2]
1
```

Therefore (2391203911756701601, 663200639532988920) is a solution of the Diophantine equation $x^2 - 13\, y^2 = 1$.

Exercise Set 2.5

1. Use the program **Diophantine** to solve the following Diophantine equations.
   a) $87x + 37y = 807$
   b) $1586x - 806y = 338$.

2. Modify the program **Diophantine** to give the general solutions to the equation $ax + by = c$, and print "no solution", if gcd(a,b) does not divide c. Find the general solutions of the equations given in problem 1.

3. Compare the program **Diophantine** and **LinearDiophantine**. Which is faster?

4. Prove by induction that if (x_1, y_1) is the solution of Pell's equation generated by the above method, then all subsequent solutions x,y satisfy
   $x+y\sqrt{d} = (x_1+ y_1\sqrt{d})^r$, for $r \geq 1$.
   For example $(8+3\sqrt{7})^2 = 127 + 48\sqrt{7}$.

5. Run the program **Pell** to find the first few solutions of the Diophantine equation $x^2 - 31\, y^2 = 1$. Find the next solution by similar calculations to the above calculations.

6. The least positive solution of the diophantine equation $x^2 - 61\, y^2 = 1$ is x= 1766319049, y = 226153980. Find another solution.

7. In general if (x,y) is an integral solution of the Diophantine equation $x^2 - d\, y^2 = n$, where d >0 is not a perfect square, and $|n| < \sqrt{d}$, then $\frac{x}{y}$ is a convergent of the continued fraction of $\sqrt{d}$. Modify the program Pell to check if x=p(n) and y=q(n) are solutions for the Diophantine equation $x^2 - d\, y^2 = n$. Find the first few soluions of the Diophantine equation $x^2 - 37\, y^2 = 5$.

2.7 Pythagorean Triples.

One of the first theorems we learn in mathematics is the Pythagorean theorem which states that a necessary and sufficient condition for a triagnle to be a right triangle is that the sum of the squares of the lengths of its shorter sides is equal to the square of the length of its longer side (hypotheneuse). This is another example of a quadratic Diophantine equation, as in order to find all right triangles with integer sides, we need to find integers x, y, and z satisfying the diophantine equation $x^2 + y^2 = z^2$. The positive integers (x, y, z) which satisfy this equation are called Pythagorean triples.

Example 2.4. The triples (3,4,5) , (6,8,10), and (5,12,13) are all Pythagorean triples since $3^2 + 4^2 = 5^2$, $6^2 + 8^2 = 10^2$, and $5^2 + 12^2 = 13^2$.

Definition 2.2. A Pythagorean triples (x, y, z) is called *primitive* if x, y, and z are pairwise coprimes.

Example 2.5. The Pythagorean triples (3,4,5) and (5,12,13) are primitive but (6,8,10) is not.

Note that any integral multiple of a primitive Pytahgorean triple is also a Pythagorean triple (show this.). Consequently, all Pythagorean triples can be found by forming integral multiples of primitive Pythagoran triples.

Unlike most nonlinear diophantine equations, it is possible to explicitly describe all the integral soluions of the equation $x^2 + y^2 = z^2$.

Theorem 2.5. The positive integers x,y,z form a primitive Pythagorean triple, if and only if there are relatively prime positive integers *m* and *n*, with m>n, and *m* is not congruent to *n* (mod 2),
such that

$$x = m^2 - n^2$$
$$y = 2mn$$

$$z = m^2 + n^2.$$

For the proof of this theorem see [Rosen].

Example 2.6. Let m = 5 and n = 2. Hence

$$x = m^2 - n^2 = 5^2 - 2^2 = 21$$
$$y = 2\,mn = 2.5.2 = 20$$
$$z = m^2 + n^2 = 5^2 + 2^2 = 29.$$

The following program produces a list of Primitive pythagorean triples for the values $m \le k$, using Theorem 2.5.

New MATHEMATICA Commands:

```
MemberQ[list, form] returns True if an element of list
   matches form, and False otherwise.

Abs[z] gives the absolute value of the real or complex
   number z.
```

Program 2.11: PythagoreanTriples

```
**********************************************************************
     PythagoreanTriples[k_]:=(
          even=Select[Range[n],EvenQ];
     Do[  mQ[j_] := MemberQ[{1},GCD[j,i]];
          s=Select[even,mQ];
               x=Abs[i^2-s^2];
               y=2 i s;
               z=i^2 +s^2;
     Do[
          Print["( ",x[[n]],",",y[[n]],",",z[[n]],")"],
               {n,Length[s]}],
                              {i,1,k-1,2}])
**********************************************************************
     PythagoreanTriples[10]
     ( 3,4,5)
     ( 15,8,17)
     ( 35,12,37)
     ( 63,16,65)
     ( 99,20,101)
     ( 5,12,13)
```

```
( 7,24,25)
( 55,48,73)
( 91,60,109)
( 21,20,29)
( 9,40,41)
( 11,60,61)
( 39,80,89)
( 45,28,53)
( 33,56,65)
( 13,84,85)
( 15,112,113)
( 51,140,149)
( 77,36,85)
( 65,72,97)
( 17,144,145)
( 19,180,181)
```

Exercise Set 2.6

1. Find the primitive Pythagorean triples for m and n ≤ 40.

2. Modify the program **PythagoreanTriples** to list all primitive Pythagorean triples x,y,z with z ≤ n.

3. Let x_n, y_n, z_n be defined recursively by $x_1=3$, $y_1=4$, $z_1=5$, and
   $x_{n+1}= 3\,x_n+2z_n+1$
   $y_{n+1}= 3\,x_n+2z_n+2$
   $z_{n+1}= 4\,x_n+3z_n+2$
   Write a program to list the triples x_k, y_k, z_k, for $k \le n$. Do you get a Pythagorean triple for all *k*?

4. Modify the program **PythagoreanTriples** to list all primitive Pythagorean triples x,y,z with y = x+1. Do you find these among the Pythagoran triples given in problem 3?

5. Modify the program **PythagoreanTriples** to find all solutions in positive integers of the Diophantine equation $x^2+ py^2 = z^2$, where *p* is a prime.

6. Modify the program **PythagoreanTriples** to find all solutions in positive integers of the Diophantine equation $w^2+x^2+y^2 = z^2$.

7. The Diophantine equation $x^2+y^2 = z^3$, has infinitely many integer solutions of the form $x = 3k^2-1$, $y = k(k^2- 3)$, $z = k^2+ 1$, $k\geq 0$. Write a program to list all solutions of this equation for *k* less than a given bound.

2.8 Sums of Two Squares.

Another problem that has interested mathematicians through out history has been the representation of integers as sums of squares. Not all integers are sum of two squares. For example m=3 or m=23 can not be represented as sum of two squares whereas 20 can be written as $20 = 4^2+ 2^2$. The question is which integers are the sum of two squares? A number of mathematicians have made important contributions to this problem including Diophantus, Fermat, Euler and Lagrange are some of them.

The following program produces a list of integers less than *n* that can be written as the sum of two squares.

Program 2.12: TwoSquares

```
*************************************************************
      TwoSquares[n_] := (m = Floor[N[Sqrt[n]]];
                r = Range[0,m]^2;
                s = n-r;  mQ[k_] := MemberQ[r,k];
                t = Select[s,mQ];
                g = Length[t]; i = (g+1)/2;
   If[g==0,Print[n," is not the sum of two squares"],
      Do[Print[  n,"  =   ",  Sqrt[t[[j]]]^"2", "   +   ",
Sqrt[t[[g+1-j]]]^"2"],
                  {j,i}]];)
*************************************************************

     TwoSquares[250]

             2    2
     250 = 15  + 5

             2    2
     250 = 13  + 9
```

The following modification prints a list of integers less than or equal to n as sum of two squares, if possible.

Program 2.13: TwoSquaresTable

```
**********************************************************************
TwoSquaresTable[f_] := (TwoSquares /@ Range[f];)
**********************************************************************
```

```
TwoSquaresTable[10]
```

```
1 = 1 + 0
2 = 1 + 1
3 is not the sum of two squares
4 = 2^2  + 0
5 = 2^2  + 1
6 is not the sum of two squares
7 is not the sum of two squares
8 = 2^2  + 2^2
9 = 3^2  + 0
10 = 3^2  + 1
```

Exercise Set 2.7

1. Get a list of integers less than or equal to 30 in terms of sum of two squares, using program **TwoSquaresTable**.
   Can you see anything in common among those positive integers which are not representable as the sum of two squares? (How about the prime integers?)
   The following theorem tell us which prime integers can be written as sum of two squares.
   Theorem 2.6. If *p* is a prime not of the form 4k + 3, it can be written as sum of two squares.

2. Verify this theorem for the first 30 prime integers of the form 4k +3, using the program **TwoSquares.**

3. Prove the following theorem.
   Theorem 2.7. If *m* and *n* are both sums of two squares then *mn* is also

the sum of two squares. Hint: Show that $(a^2+b^2)(c^2+ d^2) = (ac + bd)^2+ (ad – bc)^2$.

4. Verify the following theorem for the list obtained in problem 1.

   Theorem 2.8. A positive integer *n* is the sum of two squares if and only if each odd prime factor of *n* is of the form 4k +3 occurs to an even power in the prime factorization of *n*..

5. Use the program **`TwoSquares`** together with the theorems 2.7 and 2.8 to represent the following integers as sum of two squares.
   a) 10612420
   b) 5407969280

2.9. Sums of Three or More Squares.

Not all positive integers can be written as sum of three squares.

The following theorem classifies the integers that can be written as sum of three squares:

Theorem 2.9. Any number *not* of the form $4^r(8k +7)$ can be written as sum of three squares.

The proof of this result is hard and is due to Legendre, Drichlet and Gauss.

Sums of Four Squares.

All positive integers can be written as sum of four squares. Fermat wrote that he had a proof of this fact, although he never published it. Euler was unable to find a proof, although he made substantial progress towards a solution (The identity in problem 1, exercise set 2.8). It was in 1770 when Lagrange presented the first published proof.

All positive integers can be written as sum of four squares. The proof of this famous result which was necessary for formation of Quarternions(see [ ]), is based on the following two lemmas.

Lemma 2.1. If *p* is a prime then *p* can be written as sum of four squares.[Rosen]

The proof of this lemma uses modular arithmetic.[Rosen]

Lemma 2.2. If *m* and *n* are positive integers which are both the sum of four squares, then *mn i*s also the sum of four squares.

The proof of this lemma is simple and can be checked by MATHEMATICA, Exercise 1, exercise set 2.8.

The following program in MATHEMATICA finds the sum of four squares for any integer.

New MATHEMATICA Commands:

```
Thread[f[args], h] threads f over any objects with head h
  that appear in args.
```

Program 2.14: FourSquares

```
**********************************************************
     sq1[n1_] := (m1 = Floor[N[Sqrt[n1]]];
          r1 = Range[0,m1]^2;
          s1 = n1-r1;  mQ1[k_] := MemberQ[r1,k];
          t1 = Select[s1,mQ1];
          u1 = Thread[{t1,Reverse[t1]}];
          g1 = Length[t1]; i1 = Floor[(g1+1)/2];
          q1 = Take[u1,i1]);
FourSquares[n_] := (rg = Range[0,n]; s1 = sq1 /@ rg; m =
Floor[(n+1)/2];
 Do[l1 = Length[s1[[i]]]; l2 = Length[s1[[n-i+2]]];
          If[l1*l2 >0, Do[Do[
            Print[n," = ", Sqrt[s1[[i,j,1]]]^"2",
            " + ", Sqrt[s1[[i,j,2]]]^"2",
             " + ",Sqrt[s1[[n-i+2,k,1]]]^"2",
            " + ", Sqrt[s1[[n-i+2,k,2]]]^"2"]
                 ,{k,l2}],{j,l1}]],{i,m}])

**********************************************************
```

In this program, **sq1** is basically a modification of the program used to get the sum of two squares. **FourSquares** breaks a number into sum of two numbers and finds the sum of two squares for each of the two numbers and augments them together.

```
FourSquares[25]
```

$25 = 0 + 0 + 5^2 + 0$

$25 = 0 + 0 + 4^2 + 3^2$

$25 = 2^2 + 1 + 4^2 + 2^2$

$25 = 2^2 + 2^2 + 4^2 + 1$

$25 = 3^2 + 0 + 4^2 + 0$

Here we have all different possibilities for n = 25. Trying a number larger than 1000 takes a lot longer. For example for **`FourSquares[1345]`** we got over 15 pages of output. Here are some of them:

$1345 = 0 + 0 + 36^2 + 7^2$
$1345 = 0 + 0 + 33^2 + 16^2$
$1345 = 2^2 + 0 + 30^2 + 21^2$
$1345 = 3^2 + 2^2 + 36^2 + 6^2$
$1345 = 4^2 + 2^2 + 35^2 + 10^2$
$1345 = 4^2 + 2^2 + 34^2 + 13^2$
$1345 = 4^2 + 2^2 + 29^2 + 22^2$
$1345 = 4^2 + 4^2 + 32^2 + 17^2$
$1345 = 4^2 + 4^2 + 28^2 + 23^2$
$1345 = 6^2 + 2^2 + 36^2 + 3^2$

Try **`FourSquares[n]`**, for two different *n*'s not larger than 200. For n= 200, it takes around 43 seconds.

Sums of Any Number of Squares.

Naturally the question of sums of four squares can be generalized. Can every positive integer be written as sums of a *fixed number of squares*? We have seen the answer to this question for n= 2, 3, and 4.

Here is a program in MATHEMATICA that can find the sum of *n* squares, n ≥ 2, for any integer, if possible.

New MATHEMATICA Commands:

```
Append[expr, elem] gives expr with elem appended.

SequenceForm[expr1, expr2, ...] prints as thetextual
  concatenation of the printed forms of the expri.

Superscript[expr] prints expr as a superscript.
```

In this program, **`SumSquares,`** finds the sum of any given integer *n* in any number of squares desired *m*..

Program 2.15: SumSquares

```
*************************************************************
sq[{l_,n_}] := ( If[n==0,{{Append[l,0],0}},
         If[Length[l]==0, z=Infinity,z=Last[l]];
          m=Min[Floor[N[Sqrt[n]]],z];
          r = Range[m]; s = n-r^2;
          Thread[{Append[l,#]& /@ r,s}]]);

fsq[n_Integer] := sq[{{},n}];
fsq[s_List] := Flatten[sq /@ s,1];
doneQ[s_] := Last[s]==0;

squares[n_,m_]:= First /@ Select[Nest[fsq,n,m],doneQ];

pr[n_,s_] := (Print[
 n," = ",Infix[s,SequenceForm[Superscript[2]," + "]],
  Superscript[2]];Print[])
          SumSquares[n_,m_]:=
                          (pr[n,#]& /@ squares[n,m];);
*************************************************************
```

```
SumSquares[25,8]
```

$25 = 2^2 + 2^2 + 2^2 + 2^2 + 2^2 + 2^2 + 1^2 + 0^2$

$25 = 3^2 + 2^2 + 2^2 + 2^2 + 1^2 + 1^2 + 1^2 + 1^2$

$25 = 3^2 + 2^2 + 2^2 + 2^2 + 2^2 + 0^2 + 0^2 + 0^2$

$25 = 3^2 + 3^2 + 2^2 + 1^2 + 1^2 + 1^2 + 0^2 + 0^2$

$25 = 4^2 + 2^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2$

$25 = 4^2 + 2^2 + 2^2 + 1^2 + 0^2 + 0^2 + 0^2 + 0^2$

$25 = 4^2 + 3^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2$

$25 = 5^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2$

The advantage of this program over **FourSquares** program is that there are

no repeats and for n= 200, it takes around 24 seconds.

SumSquaresTable finds a table of *m* sums of squares for all integers less than or equal to .

Program 2.16: SumSquaresTable

```
*************************************************************
SumSquaresTable[n_,m_]:=
                        SumSquares[#,m]& /@ Range[n]
*************************************************************
```

```
SumSquaresTable[10,3]
```

$1 = 1^2 + 0^2 + 0^2$

$2 = 1^2 + 1^2 + 0^2$

$3 = 1^2 + 1^2 + 1^2$

$4 = 2^2 + 0^2 + 0^2$

$5 = 2^2 + 1^2 + 0^2$

$6 = 2^2 + 1^2 + 1^2$

$8 = 2^2 + 2^2 + 0^2$

$9 = 2^2 + 2^2 + 1^2$

$9 = 3^2 + 0^2 + 0^2$

$10 = 3^2 + 1^2 + 0^2$

Exercise Set 2.8

1. Prove the following identity using algebraic commands of MATHEMATICA .
   If $m = a^2 + b^2 + c^2 + d^2$ and $n = e^2 + f^2 + g^2 + h^2$
   then,
   $mn = (a^2 + b^2 + c^2 + d^2)\ (e^2 + f^2 + g^2 + h^2) =$
   $(ae + bf + cg + dh)^2 + (af - be + ch - dg)^2 + (ag - bh - ce + df)^2 + (ah + bg - cf - de)^2$.
   This identity is due to Euler and shows that to prove the Four Square Theorem one need only establish its validity for primes.

2. Try **`SumSquares`** and **`SumSquaresTable`** for different integers.

3. C. P. Snow, in his foreword in the book A Mathematician's Apology, tells this story of the mathematicians Hardy and Ramanujan: " It was on one of those visits that there happened the incident of the taxi-cab number. Hardy had gone out to Putney by Taxi, as usual his chosen method of conveyance. He went into the hospital room where Ramanujan was lying. Hardy, always inept about introducing a conversation, said, probably without a greeting, and certainly as his first remark: 'I thought the number of my taxi-cab was 1729. It seemed to me rather a dull number.' To which Ramanujan replied, 'No, Hardy! It is a very interesting number. It is the smallest number expressible as the sum of two cubes in two different ways'."
   a) Modify the program **`SumSquares[n_,m_]`** to express any integer *n* as the sum of *m* cubes. Verify Ramanujan's assertion.

   Hardy then asked for the smallest number which is a sum of two *fourth* powers in two different ways, but Ramanujan did not happen to know. The answer is 635,318657 which appears to have been discovered by Euler.[Silverman]

   b) Modify the program **`SumSquares[n_,m_]`** to express any integer *n* as the sum of *m* fourth powers. Find the fourth powers that sum up to 635,318,657.

   Hardy could just as easily have asked for the smallest number which is a sum of two cubes in three distinct way. In this case the answer is given in [Leech] to be 87,539,319.

   c) Use the modified program in a) to find these cubes.

4. Show that 23 is the sum of nine cubes of nonnegative integers but not the sum

of eigth cubes of nonnegative integers. Can you find another integer with this property?

Chapter 3. Algorithms of Number Theory and Prime Numbers.

3.1. Egyptian Fractions.

The ancient Egyptians some 4000 years ago had an unusual and unique way of representing fractions $\frac{a}{b}$, where a and b are positive integers with a <b as the sum of so-called *unit fractions* , of the form $\frac{1}{n}$ where n >1, such that no unit fraction was repeated. For example $\frac{4}{23} = \frac{1}{6} + \frac{1}{138}$.

There are several differences between the Egyptian system and the decimal system. First, their representation for a rational number always stops, whereas the decimal representation need not ($\frac{1}{3} = 0.3333\ldots$). Second there are many ways to represent a number in the Egyptian method, $\frac{2}{5} = \frac{1}{3} + \frac{1}{15} = \frac{1}{4} + \frac{1}{7} + \frac{1}{140} = \frac{1}{5} + \frac{1}{6} + \frac{1}{30}$, whereas there at most two representation of a number in the decimal system. $\frac{2}{5} = 0.4 = 0.39999\ldots$

There is really no advantage in doing mathematics using the Egyptian system. In fact there are many disadvantages. For instance, it not easy to even determine when one number is larger than another. (Which is larger $\frac{1}{3} + \frac{1}{11} + \frac{1}{231}$ or $\frac{1}{2} + \frac{1}{14}$?). Addition, multiplication, etc. could not be done easily. Perhaps it was their cumbersome technique for representing numbers that prevented the Egyptian from advancing in mathematics as much as the Greek did.

We will discuss an algorithm for representing fractions in Egyptian representation.

Greedy Algorithm: We can find a representation using an ancient algorithm

Given a fraction $\frac{a}{b}$, where a and b are positive integers with a < b,

i) Let c be the least integer greater than $\frac{b}{a}$ so that $\frac{1}{c} \le \frac{a}{b}$.

ii) Form $\frac{a}{b} - \frac{1}{c}$.

iii) Let d be the least integer different from c such that $\frac{1}{d} \le \frac{a}{b} - \frac{1}{c}$.

Continue the process.

Applying the greedy algorithm to the fraction $\frac{3}{7}$,

we get

i) c = least integer greater than $\frac{7}{3} = 3$. Hence $\frac{1}{3}$ is the first unit fraction.

ii) $\frac{3}{7} - \frac{1}{3} = \frac{2}{21}$. Hence $\frac{1}{11}$ is the 2^{nd} unit fraction.

iii) d = least integer greater than $\frac{21}{2} = 11$

iv) $\frac{2}{21} - \frac{1}{11} = \frac{1}{231}$. The Algorithm terminates at this point.

Therefore $\frac{3}{7} = \frac{1}{3} + \frac{1}{11} + \frac{1}{231}$.

Fibonacci in 1202 proved that the greedy algorithm always terminates in finitely many steps.

The following MATHEMATICA program uses the greedy algorithm to find the integers involved in the denominators of the Egyptian Fraction representation:

New MATHEMATICA commands:

```
Prepend[expr, elem] gives expr with elem prepended.

Ceiling[x] gives the smallest integer greater than or equal
   to x.
```

Program 3.1: EgyptianInteger

```
**********************************************************************
     EgyptianInteger[0]  := {}
     EgyptianInteger[q_] := Prepend[
   EgyptianInteger[q - 1/Ceiling[1/q]], Ceiling[1/q]];
**********************************************************************
```

In order to get a representation in the form of fractions we must use the following Format using the command **Infix** which inserts the addition symbol '+` between the fractions. The number 100 is used to give low priority to the parenthesis involved in the list obtained and hence eliminate them from the output. Eliminate '100' to see what happens.

Program 3.2: EgyptianFraction

```
**********************************************************************
     Format[efrac[x__]] := Infix[efrac[x]," + "]
          EgyptianFraction[q_]:=
          (e = efrac @@ (1/EgyptianInteger[q]);
               Print[q," = ",e]);
**********************************************************************
```

We can check the program for q = $\frac{14}{15}$.

```
EgyptianFraction[14/15]

14    1    1    1
```

$$\frac{2}{15} = \frac{1}{2} + \frac{1}{3} + \frac{1}{10}$$

Another method used for finding Egyptian fractions is an algorithm called the *splitting method*. This method begins by writing $\frac{a}{b}$ as $\frac{1}{b} + \frac{1}{b} + \ldots. \frac{1}{b}$ and then using the identity $\frac{1}{n} = \frac{1}{n+1} + \frac{1}{n(n+1)}$ to eliminate all but the first of the $\frac{1}{b}$'s. Then the new list is taken, and for each term that appears more than once, all but the first occurrence is eliminated by another use of the identity given.

Example 3.1. $\frac{3}{7} = \frac{1}{7} + \frac{1}{7} + \frac{1}{7}$

$$= \frac{1}{7} + \frac{1}{8} + \frac{1}{56} + \frac{1}{8} + \frac{1}{56}$$

$$= \frac{1}{7} + \frac{1}{8} + \frac{1}{56} + \frac{1}{9} + \frac{1}{72} + \frac{1}{57} + \frac{1}{3192}$$

Exercise Set 3.1

1. What is the greedy representation of $\frac{65}{131}$?

How do you like the denominator of the last unit fraction?

Although, the greedy algorithm terminates quickly, as we have seen, it sometimes yields ridiculously long denominators. There are much better representations of $\frac{65}{131}$ as an Egyptian fraction. For example, one can combine the representations of , say, $\frac{30}{131}$ and $\frac{35}{131}$; this yields a much shorter and smaller representation:

$\frac{65}{131} = \frac{1}{3} + \frac{1}{7} + \frac{1}{99} + \frac{1}{102} + \frac{1}{12969} + \frac{1}{93534}$.

2. Write a short program to find a shorter representation with smaller denominators of $\frac{65}{131}$ by checking all decompositions such as $65 = 20 + 35$.

   List three of the shorter representations.

3. Write a MATHEMATICA program to find an Egyptian representation of $\frac{a}{b}$ using the splitting algorithm. Compare the speed of your program with **`EgyptianFraction.`**

3.2. The Euclidean algorithm.

The greatest common divisor , gcd of two integers is the largest integer that divides each of the given integers. The Euclidean algorithm for computing the gcd is considered one of the best algorithms in mathematics. According to Knuth, it is the oldest nontrivial algorithm that has survived to the present day. It is very fast and very important; two uses are

(1) Writing a rational number in lowest terms.

(2) Finding whether or not two integers are relatively prime.

MATHEMATICA has a built-in GCD function which is very fast.

```
GCD[296913240,8754867810]//Timing
```

{0.0166667 Second, 1170}

Exercise Set 3.2

1. Prove the following lemma.

Lemma 3.1. Two numbers *m* and *n* are relatively prime (are called coprimes) if and only if gcd(m,n) = 1, For simplicity we can write (m,n) = 1.

2. Find the GCD of two very large odd numbers (around 50 digits- Use scientific notation) and observe the timing. Don't be surprised if they are relatively prime. If they are, try two other numbers. Continue until you get two that are not relatively prime. If you are not having any luck write a simple program to generate two random numbers and check if they are relatively prime. Keep track of the numbers generated until you get the first two that are not relatively prime. You can use the command `Random`.

The above experiments lead us to the following question.

Question: What is the probability that two numbers chosen at random are coprime?

Answer: For a given *n* we can consider all pairs of numbers *a* and *b* with $1 \le a, b \le n$; there are clearly n^2 such numbers. If c(n) denotes the number of the coprimes, then $\frac{c(n)}{n^2}$ is clearly the probability a randomly chosen pair of numbers between *a* and *n* are coprime.

Theorem 3.1. $\lim_{n\to\infty} \frac{c(n)}{n^2} = \frac{6}{\pi^2} = 0.607927$

The proof of this theorem can be found in [Rademacher]. The proof is difficult and

relies on Reimann's □-function and the fact that $1 + \frac{1}{4} + \frac{1}{9} + ... = \sum_{r÷=÷1}^{\infty\square} \frac{1}{r^2} = \frac{\square^2}{6}$.

The following MATHEMATICA program can generate *n* randomly chosen set of numbers and finds their gcd and then finds the probability of the numbers being coprimes with in the set of numbers chosen.

New MATHEMATICA Command:

```
Random[type, range] gives a pseudorandom number of the
   specified type, lying in the specified range.
```

Program 3.3: rand

```
**************************************************************
     rand[n_] :=( k=0;
          Do [ t=Table[Random[Integer, {1,1000}],{2}];
          g= GCD[First[t],Last[t]];
               If[ pr==1, Print[t,"**** GCD=",g]];
                    If[g==1, k=k+1], {i,n}];
                         If[pr==1, Print["Total =",k]];
                         Print["Probability =",N[k/n]];)
**************************************************************
```

```
     pr=1;rand[20]
```

**Remark:If you put pr=1, it prints the output.*
If you put pr=0, it doesn't .

```
     {282, 232}**** GCD=2
     {465, 653}**** GCD=1
     {894, 591}**** GCD=3
     {527, 637}**** GCD=1
     {975, 346}**** GCD=1
     {443, 672}**** GCD=1
     {769, 101}**** GCD=1
     {238, 622}**** GCD=2
     {401, 316}**** GCD=1
     {455, 547}**** GCD=1
     {966, 650}**** GCD=2
     {895, 146}**** GCD=1
     {49, 444}**** GCD=1
```

```
{781, 487}**** GCD=1
{314, 481}**** GCD=1
{279, 6}**** GCD=3
{372, 493}**** GCD=1
{116, 812}**** GCD=116
{232, 387}**** GCD=1
{807, 933}**** GCD=3
Total =13

Probability =0.65
```

pr=0;rand[500]

Probability =0.616

Exercise Set 3.3

1. Use the program **rand** to find the probability of 1000, and 2000 randomly chosen pair of numbers being coprimes. Does the probability get closer to 0.607927?

2. Modify the program to print and count the integers a ≤ n, such that *a* and *n* are coprime for any given *n*.

   The count can be given by the *Euler phi function* $\phi(n)$ which denotes the number of positive integers less than *n* and relatively prime to *n*.
   MATHEMATICA has a built-in **EulerPhi** function

   EulerPhi[549]

   360

3. Modify the program used in the problem 2, using **EulerPhi** to count the number of integers less than *n* and relatively prime to *n*.

   You can use the following MATHEMATICA program to get a list for the probabilities that a pair chosen from {1,200} {1,400},...,{1,2000} is relatively prime.
   Note that the denominator n + Binomial[n,2] is used because that is how many

pairs there are from {1,...,n} including pairs of the form (i,i).

```
Table[(Plus @@ EulerPhi /@ Range[n])/
(n + Binomial[n, 2]),{n, 200,2000,200}]//N
```

Does the list approach 0.607927?

4. The **EulerPhi** function has a number of remarkable properties. Investigate these properties for a few values of your choice.
   a) If d_1 and d_2 are coprimes then $\phi(d_1 d_2) = \phi(d_1)\,\phi(d_2)$.
   b) If p^n is a prime power $\phi(p^n) = p^n - p^{n-1}$ (This is easy to prove by pen and paper).
   c) The sum of the numbers $\phi(x)$, where *x* is a divisor of *d*, is equal to *d* itself.
      For example: For d=18, $\phi(18)=\phi(1)+\phi(2)+\phi(3)+\phi(6)+\phi(9)+\phi(18)=$
      $= 1 +1 +2 +2 +6 +6 = 18.$
   d) $x^{\phi(d)} \equiv 1 \pmod d$ for any x coprime to d.

3.3. Prime Numbers.

There has always been a lot of interest in the Prime numbers since the ancient times. For example it was known to the ancient Greek mathematician Euclid, that the number of primes is infinite.

Theorem 3.2. There are infinitely many primes.

The proof of this theorem given by Euclid is so simple and beautiful.

Suppose that there are only a finite number of primes $p_1, p_2,....p_n$.

Now consider $q= p_1.p_2.....p_n+1$.

It is clearly not divisible by any of the primes $p_1, p_2,....p_n$ (Why?). Consequently the prime divisors of *q* provide new prime numbers, contradicting our assertion that $p_1, p_2,....p_n$ is a complete list.

Another Greek scholar-mathematician Eratosthenes (276-194 B.C.E.) had found a procedure for finding all the primes less than or equal to a given integer *n*. This procedure is called the *sieve of Eratosthenes*.

Sieve of Eratosthenes Algorithm.

Given a composite integer *n*, we first notice that *n* has a prime factor not exceeding $\sqrt{n}$. If we write n = a b, where *a* and *b* are integers with $1< a \le b < n$. We must have $a \le \sqrt{n}$, since otherwise $b \ge a > \sqrt{n}$ and $a\,b > \sqrt{n}\,.\sqrt{n} = n$. We now check each integer less than n

for divisibility by the primes less than $\sqrt{n}$. We first cross out multiples of 2. Next we cross out those integers remaining that are multiples of 3. Then all multiples of 5 not yet crossed off. Continuing in this fashion we obtain a list of the prime numbers between 1 and *n*..

Try this method out with paper and pencil for n=100.

We can write a program in MATHEMATICA implementing Eratosthenes' method.

New MATHEMATICA Command:

```
Fold[f, x, {a,b, ...}] gives the last element of {x, f[x, a],
   f[f[x, a], b], ...}.
The symbol || represents or.
```

We make a list *l* of the numbers {1, 2, 3, ..., n}. The function `emult[l_,k_]` removes all multiples of *k*, except for *k* itself, from the list *l*, where *k* runs over the set {2, 3, ..., $\sqrt{n}$ }. The function `Fold[emult,r,m]` applies `emult` to *r* and *m* in an iterative way.

Program 3.4: Eratosthenes

```
*****************************************************************
     Eratosthenes[n_] := (r = Range[n];
                          m = Range[2,Floor[N[Sqrt[n]]]];
 emult[l_,k_] := Select[l,(Mod[#,k]!=0 || # == k )&];
     Fold[emult,r,m])
*****************************************************************
     Eratosthenes[100]//Timing

{2.58333 Second,{1,2,3,5,7,11,13,17,19,23,29,31,37,41,
     43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97}}.
```

As you can see, although, the sieve of Eratosthenes produces all primes less than or equal to a fixed integer, it is quite inefficient.

One could improve the program by removing all multiples of *k*, except for k itself, from the list *l*, where *k* runs over the set {2, 3, ..., $\sqrt{n}$ }and *k* is not a composite number.

We get the following program:

Program 3.5: ImprovedEratosthenes

```
*****************************************************************
     ImprovedEratosthenes[n_] :=
```

```
     (r = Range[2,n]; m = Range[2,Floor[N[Sqrt[n]]]];
emult[l_,k_] := Select[l,(Mod[#,k]!=0 || # == k )&];
       While[Length[m] > 0,
       k = First[m]; r = emult[r,k];
              m = emult[Rest[m],k]]; r)
*************************************************************
```

```
     ImprovedEratosthenes[100]//Timing
```

```
{1.8 Second, {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41,
43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97}}.
```

To determine whether a particular integer *n* is prime in this manner, it is necessary to check *n* for divisibility by all primes not exceeding $\sqrt{n}$.

The divisibility check is also quite inefficient. There are of course better methods for deciding whether or not an integer is prime. Fortunately for us, MATHEMATICA has a built-in function **PrimeQ** which can determine whether or not a number is prime and it works on arbitrarily large integers very efficiently.

```
     PrimeQ[2^517 +1]//Timing
     {6.48333 Second, False}

     PrimeQ[10^40 +2059]//Timing
     {7.33333 Second, True}
```

Another useful command is **Prime[n]** which gives the *n*th prime.

```
     Prime[10^9]
     22801763489.
```

Exercise Set 3.4

1. Write a program similar to **Eratosthenes** to determine whether a given

integer is a prime, using the divisibility check.
Check your program for small numbers.

2. For a given integer *n*, use **PrimeQ**, to write a program to find the first prime larger than *n*.
   Use your program to find a prime number larger than 10^{100}.

3. We can write a one line program to generate a list of the primes between 1 and n, using **Select** and **PrimeQ**, namely
   PrimesUpTo[n_]:=Select[Range[n],PrimeQ]
   Compare the Timing of this program with that of the program **ImprovedEratosthenes.**

4. It is unknown whether there are infinitely many primes of the form n^2+1 where *n* is a positive integer. Modify the program **PrimesUpTo** to find the first 100 primes of the form n^2+1.

5. Use a simple program to show that
   a) The polynomial $x^2 - x + 41$ is prime for all integers *x* with $0 \le x < 40$. Is it a prime for x=41?
   b) The polynomial $x^2 - 79x + 1601$ is prime for all integers *x* with $0 \le x < 79$. Is it a prime for x=80?

   Both of these polynomials were found by Euler. It is known that no *polynomial* with integral coefficients can be prime for all *x*. Can you find a formula f(x) which yields only prime numbers for x=1, 2, 3,...?

6. Modify the program **PrimesUpTo** to compute the number of primes lying in an interval between *m* and *n*.
   Test your program for the following intervals.
   a) 10^3 to 10^3+10^4
   b) 10^4 to 10^4+10^4
   c) 10^5 to 10^5+10^4
   d) 10^6 to 10^6+10^4
   e) Based on you observation can you make a conjecture about the distribution of primes lying in an interval as the end-points of the interval

get larger?

3.4. The Distribution of the Primes.

As you can see from the part e) of the problem 6, exercise set 3.4, it appears that there are less primes in the intervals of the same length as the end-points of the interval get larger.

Another words it seems that the gaps between consecutive primes widens as we proceed further up the list of primes. The question is how big can these gaps become?

The answer to this question is that the gap between consecutive primes can be arbitrarily large.

Theorem 3.3. For any positive integer n, there are at least n consecutive composite positive integers.

Proof. Consider the n consecutive integers, (n+1)! +2, (n+1)! +3,...,(n+1)! +n+1. We know that $j \mid (n+1)!$, for $2 \le j \le n+1$. Therefore $j \mid (n+1)!+j$. Hence, these n consecutive integers are composite.

Even though the gaps between consecutive primes is arbitrarily large, occasionally we get primes separated by a single non-prime, i.e. p and p+2, such as 9933611 and 9933613. Such primes are called the *twin* primes. The second question one could ask is whether there are infinity many pairs of twin primes? This problem is much deeper - indeed unsolved. What is your conjecture?

Exercise Set 3.5

1. Write a program which prints out all the twin primes p, p+2 in any given interval. Find all twin primes less than 10000.

2. Prove by pen and paper that there are no "prime triplets", i.e. p, p+2, p+4, other than 3, 5, 7.

3. "Prime quadruplets" are four primes of the form p, p+2, p+6, and p+8, like 11,13,17,19; or 191, 193, 197, 199. Modify your program of the problem 1, to print out all the "Prime quadruplets" in any given interval. Find all the Prime quadruplets that lie between 90000 and 100000.

4. The "Goldbach's Conjecture" is that Every even positive integer greater than two can be written as the sum of two primes. For instance 100 = 3+97 = 11 + 89 = 17 + 83 = 29+71 = 41+ 59. Write a program to verify the Goldbach's

Conjecture for all even integers less than 10000.

5. In 1937 Vinogradov proved that every odd positive integer beyond some point (he did not determine the point), is the sum of three primes. It is also known that each odd positive integer less than 100,000 is the sum of three primes. Can you find an odd integer larger than 100,000 which can not be written as the sum of three primes?

6. Some primes are of the form n^2-4 for some positive integer *n*. For instance, $5 = 3^2-4$.
   a) Write a simple MATHEMATICA routine to list 30 primes of this form.
   b) Make a conjecture based on your observation.
   c) Prove your conjecture.

7. If *n* is a positive integer larger than 1, is there always a prime between *n* and 2*n*? Write a simple MATHEMATICA routine to experiment with 50 large numbers n, $n \geq 1000$. Make a conjecture.

8. For each integer n which is larger than 1, we can form the product
$$B_n = \left(1-\frac{1}{(p_2-1)^2}\right)\left(1-\frac{1}{(p_3-1)^2}\right) \ldots \left(1-\frac{1}{(p_n-1)^2}\right),$$
where p_n is the *n*th prime; $p_2=3$, $p_3=5$, and so on. Make a table listing the value of B_n, for the 50 primes.
   a) What number is B_n approaching to?
   b) Prove that for any *n*, B_n is larger than $\frac{1}{2}$.

9. The function $\pi(n)$, where n is a positive integer, denotes the number of primes not exceeding *n*. The command **PrimePi[x]** gives the number of primes less than or equal to *x*.
   Write a program and examine the ratio $\frac{\pi(n)}{n}$ as *n* gets larger and larger. In the language of limits, can you make a conjecture about $\lim_{n\to\infty} \frac{\pi(n)}{n}$.

We know that there are infinitely many primes, but can we estimate how many primes there are less than n for a very large given number *n*. One of the most famous theorems of number theory, and of all mathematics, answers this question.

Theorem 3.4. The Prime Number Theorem. The ratio of $\pi(n)$ to $\frac{n}{\log n}$

approaches one as n grows without bound. i.e. $\lim_{n\to\infty} \frac{\pi(n)}{n/\log n} = 1$.

The Prime Number Theorem was first conjectured in 1792 by a 15-year-old schoolboy. The schoolboy in question was Carl Friedrich Gauss, one of the greatest mathematicians to have ever lived, but it was not proved until 1896, when a French mathematician J. Hadamard and a Belgian mathematician C.J. de la Vallee-poussin produced independent proofs. The proof is quite complicated and uses complex numbers and calculus. It can be found in [Hardy and Wright]. In 1948, Selberg and Erdos independently discovered a proof that requires neither complex numbers or calculus.

The Prime Number Theorem tells us that $\frac{n}{\log n}$ is a good approximation to $\pi(n)$ when n is large. It has been shown that an even better approximation to $\pi(n)$, when n is large is given by

$$li(n) = \int_2^n \frac{dt}{\log t}$$

(where $\int_2^n \frac{dt}{\log t}$ represents the area under the curve $y = \frac{1}{\log t}$, and above the t-axis from t=2 to t=n.)

In order to compare the three values for $\pi(n)$, $\frac{n}{\log n}$ and $li(n)$, we can use MATHEMATICA to produce a table.

New MATHEMATICA Commands:

```
Round[x] gives the integer closest to x.

NIntegrate[f,{x xmin, xmax}] gives a numerical approximation
   to the integral of f with respect to x over the interval
   xmin to xmax.

TableForm[list] prints with the elements of list arranged in
   an array of rectangular cells.


Transpose[list] transposes the list.

TableHeading is an option for TableForm and MatrixForm which
   gives the labels to be printed for entries in each
   dimension of a table or matrix.
```

Program 3.6: RatioTable

```
**********************************************************
     RatioTable[x_]:=Round[N[x/Log[x]]];
          li[x_]:=Round[NIntegrate[1/Log[t],{t,2,x}]];
               g[x_]:= 10 x;
               l=NestList[g,1000,6];
     TableForm[Transpose[
          {l,Map[PrimePi, l],Map[RatioTable,l],Map[li,l]}],
                    TableHeadings ->{Automatic,
                    {"x","p(x)","x/log(x)","li(x)"}}]
**********************************************************
```

	x	p(x)	x/log(x)	li(x)
1	1000	168	145	177
2	10000	1229	1086	1245
3	100000	9592	8686	9629
4	1000000	78498	72382	78627
5	10000000	664579	620421	664917
6	100000000	5761455	5428681	5762209
7	1000000000	50847534	48254942	50849234

As we can see from the table, when the value of x is large, x/log(x) is a good approximation for $\pi(x)$ and *li*(x) is an excellent approximation. In order to evaluate $\pi(x)$ for very large numbers such as $x=10^{15}$, don't rush to your computer with **PrimePi [10^14]** . You'll have a long wait ! It can be approximated by

```
Round[N[10^14/Log[10^14]]]
3102103442166.
```

A better approximation is

```
Round[NIntegrate[1/Log[t],{t,2,10^14}]]
3204942065679.
```

These calculations were done in seconds. The actual value of $\pi(10^{14})$ is 3204941750802. For other approximations see [Wagon] and [Vardi].

3.5. Mersenne Primes.

As we have seen in the problem 5 of exercise set 4.4, there are formulas *f(n)* which yield primes for some values of n but it is known that no *polynomial* with integral coefficients can be prime for all *n*. The question is then are there more complicated functions that can produce only primes. If we move away from polynomials and allow ourselves the luxury of exponential functions there seems more hope of finding such a function. In fact the famous mathematician Fermat considered numbers of the form 2^n+1. He then noticed that for such a number to be prime $n=2^m$ for some m. Therefore, he made the following conjecture: The numbers of the form $F_m= 2^{2^m} + 1$ (which are referred to as Fermat numbers) are primes. You can easily check that this is so for m=1, 2, 3, 4. In 1732, however, Euler showed that the fifth Fermat number $F_5= 2^{32}+1$ was composite. You can easily check this by using MATHEMATICA

```
FactorInteger[2^32 +1]
{{641, 1}, {6700417, 1}}
```

You can try to see if F_6, or F_7 are prime numbers. Can you find any Fermat number that is prime? In fact, no prime F_m has been found beyond F_4.

A far more successful source of primes are the integers of the form $M_p= 2^p - 1$ for *p* prime. These numbers were first discovered by Marin Mersenne in 1644 and therefore are named after him. You can check that M_p is prime for p = 2, 3, 5, 7 but M_{11} is composite. Therefore it is not true that M_p is prime for all prime *p*. Nevertheless the integers M_p are a source of very large primes. For example $2^{11213}-1$ is a prime. Don't try to use MATHEMATICA to determine if this number is prime as it has 3375 digits. This number is far from being a record. For instance M_{216091} is also a prime number which was discovered by Slowinski in 1985. More information on Mersenne primes can be found in [Slowinski].

It is interesting to know that in 1772 Euler proved that $2^{31}-1$ was a prime number and it was the largest known prime until over a century later in 1875 Lucas showed that $2^{127}- 1$ is a prime. He also proved that $2^{67}- 1$ is not prime as conjectured by Mersenne.

Lucas showed this using the following test.

Theorem 3.5. Lucas-Lehmer primality test.

Define a sequence of integers S_k by $S_1 = 4$ and $S_k = S_{k-1}^2 - 2$ for $k > 1$. Then the Mersenne number $M_p = 2^p - 1$ is prime if and only if $M_p \mid S_{p-1}$.

The proof of this theorem can by found in [Hardy and Wright] or [Knuth]. Since the test only concerns the divisibility of S_{p-1} by M_p clearly we can define a sequence of integers with $0 \le L_k < M_p$ by

$$L_1 = 4, L_k \equiv L_{k-1}^2 - 2 \pmod{M_p},$$

If $L_{p-1} = 0$, then M_p has passed the Lucas-Lehmer primality test and is indeed a prime number.

We can write a simple MATHEMATICA program to verify this assertion using Lucas-Lehmer primality test.

Program 3.7: LucasTest

```
*************************************************************
     LucasTest[p_]:=Module[{f},f[x_]=Mod[x^2-2,2^p-1];
                          NestList[f,4,p-2]]
*************************************************************
```

We can get a list of all L_k's, for the 7*th* Mersenne number, using

```
LucasTest[7]
```

```
{14, 67, 42, 111, 0}
```

As we can see from the output, L_6 is indeed 0 and therefore $M_7 = 2^7 - 1 = 127$ is a prime number. However the following output shows that $M_{11} = 2^{11} - 1$ is not prime.

```
LucasTest[11]
```

```
{14, 194, 788, 701, 119, 1877, 240, 282, 1736}
```

The Lucas-Lehmer primality test can be performed quite rapidly. In fact it is possible to determine whether M_p is prime in $O(p^3)$ bit operations, since the Lucas-Lehmer test requires $p-1$ squaring modulo M_p, each requiring $O((\log M_p)^2) = O(p^2)$.

If we are only interested in L_{p-1}, which tells us that M_p is prime, we can use the command `Nest` to implement the Lucas-Lehmer test.

New MATHEMATICA Command:

`Nest[f, expr, n]` gives an expression with f applied n times to expr.

Program 3.8: LastOne

```
**************************************************************
      LastOne[p_]:= Module[{f},f[x_]=Mod[x^2 -2,2^p-1];
                        Nest[f,4,p-2]]
**************************************************************
```

To find whether $M_{61}= 2^{61}–1$ is prime, and timing the calculation:

```
LastOne[61]//Timing
```

{0.983333 Second, 0}

In fact this calculation is even faster that the build-in function `PrimeQ`.

```
PrimeQ[2^61-1]//Timing
```

{2.53333 Second, True}.

3.6. Mathematical Excursion.

The mathematician Stanislaw Ulam noticed a rather remarkable fact about the primes. First he numbered the points on a grid in a spiral as shown in the figure below.

```
·37  ·36  ·35  ·34  ·33  ·32  ·31
·38  ·17  · 16 ·15  ·14  ·13  ·30
·39  ·18  · 5  ·4   ·3   ·12  ·29
·40  ·19  · 6   1   ·2   ·11  ·28
·41  ·20  · 7  · 8  ·9   ·10  ·27
·42  · 21 ·22  ·23  ·24  ·25  ·26
·43  ·44  ·45  ·46  ·47  ·48  ·49  ·50
```

Fig 1

Then he marked the prime numbers and left out the composite numbers as shown below.

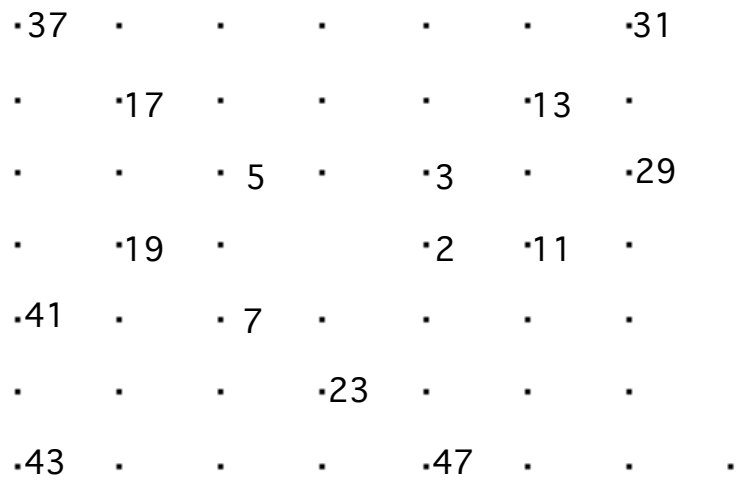


Fig 2

Amazingly he noticed a pattern. Primes tend to cluster in straight lines!

In order to write a MATHEMATICA program to plot the points in the plane corresponding to prime integers. The first thing we must do is to find a formula which finds the point on the plane corresponding to a given integer. As you can see from Fig 1, we can identify 1 with (0,0), 2 with (1,0), 3 with (1,1). If we continue in this manner, we have 17 is identified by (-2,2) and so on. Given *n*, we must find the corresponding point on the plane given by (a,b). The following algorithm does the trick.

1. Given n, find an integer c such that $(2c-1)^2 < n \le (2c+1)^2$.
2. Let $m=n-(2c-1)^2$.
3. Find an integer *k* such that $2\,c\,k < m \le 2\,c\,(k+1)$.
   (You can see that *k* can only take the values 0, 1, 2, or 3.)
4. Let $l = m - 2ck$.
5. We have the following table depending on *k*.

k	Point {a,b}
0	{c, l–c}
1	{c – l, c}
2	{–c, c–l}
3	{l–c,–c}.

To see how this algorithm works, we can do an example by hand.

1. Given n=17, we find that c = 2, since $9 < 17 \le 25$.
2. m = 17 – 9 = 8.

3. We find that k = 1, since $4 < 8 \leq 8$.
4. $l = 8 - 4 = 4$.
5. Since k = 1, the corresponding point to n = 17, is {a,b} = {2 –4, 2} = { –2, 2}.

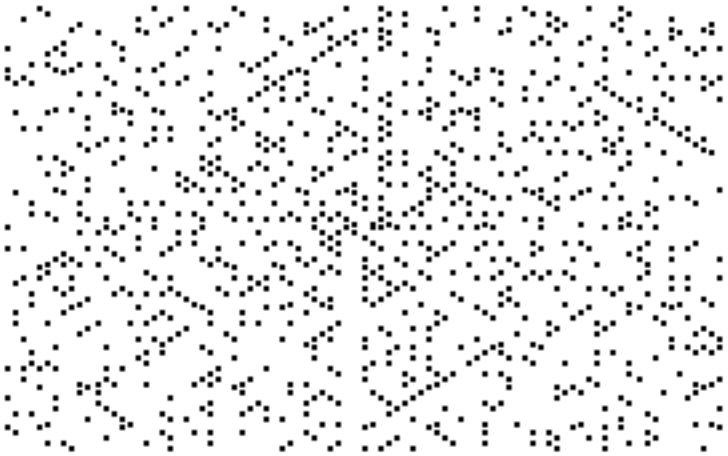
Remark: In the first part of the program, we define a function **PrimeGrid** which finds the point on the plane that corresponds to integer*n*.. Then we create a list of the first 1000 primes which we call primes and finally we map **PrimeGrid** onto primes to find a list of the corresponding prime points on the plane. We use **ListPlot** to plot these points.

New MATHEMATICA Command:

```
Switch[expr, form1, value1, form2, value2, ...] evaluates
   expr, then compares it with each of the formi in turn,
   evaluating and returning the valuei corresponding to the
   first match found.
```

Program 3.9: PrimeGrid

```
**************************************************************
     PrimeGrid[n_]:=(
          c = Min[
Floor[N[(Sqrt[n] +1)/2]],Ceiling[N[(Sqrt[n] -1)/2]]];
          m =n-(2c -1)^2;
k = Min[Floor[N[m/( 2 c)]], Ceiling[N[m/(2 c) -1]]];
          l= m - 2 c k;
Switch[k,0,{c,l-c},1,{c-l,c},2,{-c,c-l},3,{l-c,-c}];
primes=Table[Prime[i],{i,1000}];
     lis=Map[PrimeGrid,primes];
          ListPlot[lis,Axes->None]
**************************************************************
```



For a more elaborate approach see [Abbott].

Exercise Set 3.6

1. Write a program to find the proportion of values of *n* for which the Euler formula $f(n) = n^2 - n + 41$ yields primes for $0 \le n \le 1000$.

2. Write a program which checks and prints out all the Mersenne primes M_p, for *p* ranging over the first 100 primes. How many do you get? Are you surprised?

3. Use the program **LucasTest** to determine whether M_{127} is prime. Can you find the next Mersenne prime larger than M_{127}?

   Compare the calculation time of **PrimeQ** and **LastOne** for some of the Mersenne numbers. Which is faster?

4. Use the Plot command to graph $\pi(x)$, $\frac{x}{\log(x)}$ and *li*(x), for $1 \le x \le 50000$, on the same coordinate system. Use different colors or different thickness to distinguish between the graphs.

5. Modify the program **PrimeGrid** to investigate whether or not the same phenomena happens with the prime numbers between 10000 and 11000.

6. Modify the program **PrimeGrid** to plot the first 80 primes generated by the equation $g(n) = n^2 - 79\,n + 1601$. Do they eventually lie on the same line? Investigate this for the other primes it generates.

7. Write a program to list the first 10 Mersenne primes. How large is the 10th one? How many digits?

8. Use the command `Random[Integer,{min,max}]` to write a routine to randomly input very large numbers (around 25 digits). Count the number of inputs until you get a prime number? Is the number of inputs larger than what you thought to be?

9. Hardy and Littlewood in 1923 made the following conjecture which is still unproved.

   Twin Prime Conjecture: The number of pairs of twin primes less than n is approximately $\frac{1.32n}{(\ln n)^2}$.

   Use the routine of the problem 1, exercise set 3.5, to verify the Twin Prime Conjecture for n= 1000, 5000, and 10,000.