Discrete Mathematics
Homework 3

Ethan Bolker

November 3, 2014

Due: ???

We're studying number theory (leading up to cryptography) for the next few weeks. You can read Unit NT in Bender and Williamson.

This homework contains some programming exercises, some number theory computations and some proofs. The questions aren't arranged in any particular order. Don't just start at the beginning and do as many as you can – read them all and do the easy ones first.

I may add to this homework from time to time.

Exercises

1. Use the Euclidean algorithm to compute the greatest common divisor d of 59400 and 16200 and find integers x and y such that

   $$59400x + 16200y = d$$

   I recommend that you do this by hand for the logical flow (a calculator is fine for the arithmetic) before writing the computer programs that follow.

   There are many websites that will do the computation for you, and even show you the steps. If you use one, tell me which one.

2. An interesting class of examples.

   (a) Use the Euclidean Algorithm *by hand* to find an integral solution to the equation

   $$55x + 89y = 1$$

   (b) Your answer to the previous question should suggest a nice identity about the Fibonacci numbers. State it, then prove it by induction. (Look up "Fibonacci numbers" if you have to.)

3. Logarithmic time

   (a) Prove that if you carry out *two steps* in the Euclidean algorithm for $\gcd(a, b)$ with $a > b$ the remainder is less than $a/2$.

   (b) Prove that the Euclidean algorithm takes at most $2\log_2(a)$ steps. Show that is at most five times the number of decimal digits of a.

4. Computer programs

   Since CS110 is a prerequisite for this course, you should all be able to write these programs. But for some of you your programming skills are so rusty that polishing them up so you can answer this question isn't worth the time. If that's the case, just say so and skip it.

   You may do this in any language you choose (there are easier ones than Java). You might even be able to write it in Excel without macros. I'd enjoy seeing that.

   (a) Write a program (function, method, procedure) that accepts two integers as input and produces their gcd as output.

   A well written program will do the right thing when the input values are any integers – positive, negative or zero. The only case that might require special treatment is $\gcd(0, 0)$. There is no right answer then. Just make sure your program doesn't crash.

- If possible, the function that does the computation should not do any printing – it should return the answer. Then write a program that calls that function and prints the output.
  Printing is the responsibility of the calling program. If possible, the calling program should get the integer input values from the command line, or from stdin (System.in in Java). (Of course that's possible. But if your programming skills are so rusty that it's really difficult, don't spend time on it.)
- It's really easy to find solutions to this problem on the web. I'd rather you wrote your own, but won't insist. If you do get one from the web you must acknowledge and understand the source and run the program to test it.
- *Instrument* your program so that when a reporting flag is set it prints the number of iterations/recursions. Use your instrumentation to check the assertion in Problem 3b.
- Submit hard copy of your program. If possible, do that in this LaTeX document using the `listings` package. This is a particularly useful part of the homework for cs students.

(b) Improve your solution to the previous problem so that your program both finds the gcd of its input values and also finds the coefficients for a linear integral combination of the inputs that produces the gcd.

I'd still rather your function do no printing, but that's harder to arrange now that there are three integer outputs rather than just one. Do that if you can, but if you can't don't worry.

(c) If you can, write both programs so that they run in constant space. In particular, no recursive calls, since that would create a logarithmic number of stack frames.

This is easier for the first program than the second.

5. Calculate

$$54^{100} \pmod{101}$$

using the fast **Right-to-left binary method** described at `http://en.wikipedia.org/wiki/Modular_exponentiation`.

Use a calculator along the way.

If you don't like the wikipedia discussion you can find lots of others by googling `fast modular exponentiation`.

This is the computation I botched at the end of lecture.

Note: you should get 1 as the answer – that's Fermat's Little Theorem.

6. Large primes.

Note: you can look up the answers to almost all these questions. In fact I've asked you to do that for the last few. I'd rather you didn't at the beginning – you'll learn more that way.

(a) Prove

Theorem 1. *If $2^n - 1$ is prime then n is prime.*

Hint: If $n = ab$ is not prime then $2^n = (2^a)^b$. Then use a finite geometric series.

Primes of this form are called "Mersenne primes". The first few are $3 = 2^2 - 1$, $7 = 2^3 - 1$, $31 = 2^5 - 1$ and $127 = 2^7 - 1$.

(b) State the *converse* of Theorem 1. Then show that it is false.

Hint. Try to continue the list above in the obvious way.

(c) Look up some information about how the largest known prime has increased over time. Argue from the data that the logarithm of the logarithm of the largest known prime is growing. That means the largest known prime is growing much faster than exponentially!

(d) Check Chris Caldwell's Prime Pages at `http://primes.utm.edu/`. Read about GIMPS at `http://www.mersenne.org/`. Tell me something you found particularly interesting (not just something from the first page).

7. Show that $2^{340} \equiv 1 \pmod{341}$ even though 341 is not prime. What is the connection between this result and the converse of Fermat's Little Theorem?

8. Eratosthenes' RSA public key is

$$n = 10967535067$$
$$e = 1051$$

Archimedes encrypts his message with this key and sends Eratosthenes

$$C = 1963501580$$

(a) Break the encryption. (Use any tools you like; tell me how you did it.)

(b) Why is the message appropriate?

(c) What is each of these Greeks famous for?

```
% Math 320 hw3
%
\documentclass{article}
\pagestyle{empty}
\usepackage[textheight=10in]{geometry}

\usepackage{amsmath}
\usepackage{amsthm}
\usepackage{listings}
\usepackage{graphicx}
\usepackage{verbatim}

\usepackage{hyperref}

\newtheorem{theorem}{Theorem}

\newcommand{\coursehome}
{http://www.cs.umb.edu/~eb/320}

\title{Discrete Mathematics \\
Homework 3
}
\author{Ethan Bolker}
%\date{September 1, 2014}


\newcommand{\ZZ}{\mathbb{Z}}

%create (mod n) macro
\newcommand{\mm}[1]{%
\ensuremath{(\text{mod } #1)}}

\begin{document}

\maketitle

\noindent
Due: ???

We're studying number theory (leading up to cryptography) for the next
few weeks. You can read Unit NT in Bender and Williamson.

This homework contains some programming exercises, some number
theory computations and some proofs. The questions aren't arranged in
any particular order. Don't just start at the beginning and do as many
as you can -- read them all and do the easy ones first.

I may add to this homework from time to time.

\section*{Exercises}

\begin{enumerate}

\item Use the Euclidean algorithm to compute the greatest common
  divisor $d$ of
$59400$ and $16200$ and find integers $x$ and $y$ such that
\begin{equation*}
59400x + 16200y = d
\end{equation*}
```

```
I recommend that you do this by hand for the logical flow (a
calculator is fine for the arithmetic) before writing the computer
programs that follow.

There are many websites that will do the computation for you, and even
show you the steps. If you use one, tell me which one.

\item An interesting class of examples.

\begin{enumerate}

\item Use the Euclidean Algorithm \emph{by hand} to find an integral
  solution to the equation
%
  \begin{equation*}
55x + 89y = 1
  \end{equation*}

\item
Your answer to the previous question should suggest a nice identity
about the Fibonacci numbers. State it, then prove it by
induction. (Look up ``Fibonacci numbers'' if you have to.)

\end{enumerate}
\item Logarithmic time

\begin{enumerate}

\item Prove that if you carry out \emph{two steps} in the Euclidean
  algorithm  for $\text{gcd}(a,b)$ with $a >b$ the remainder is less
  than $a/2$.

\item \label{logtime} Prove that the Euclidean algorithm takes at most
  $2\log_{2}(a)$   steps. Show that is at most five times the number
  of decimal digits  of $a$.

\end{enumerate}


\item Computer programs

Since CS110 is a prerequisite for this course, you should all
be able to write these programs. But for some of you your programming
skills are so rusty that polishing them up so you can answer this
question isn't worth the time. If that's the case, just say so and
skip it.

You may do this in any language you choose (there are easier ones than
Java). You might even be able to write it in Excel without macros. I'd
enjoy seeing that.

\begin{enumerate}
\item  Write a program (function, method, procedure) that accepts two
  integers as input and produces their gcd as output.

A well written program will do the right thing when the input values
are any integers -- positive, negative or zero. The only case that
might require special treatment is $\text{gcd}(0,0)$. There is no right
answer then. Just make sure your program doesn't crash.
```

```
\begin{itemize}

\item
If possible, the function that does the computation should not do any
printing -- it should return the answer. Then write a program that
calls that function and prints the output.

Printing is the responsibility of the calling program. If possible, the
calling program should get the integer input values from the command
line, or from \lstinline!stdin! (System.in in Java). (Of course that's
possible. But if your programming skills are so rusty that it's really
difficult, don't spend time on it.)

\item  It's really easy to find solutions to this problem on the
web. I'd rather you wrote your own, but won't insist. If you do
get one from the web you must acknowledge and understand the source
and run the program to test it.

\item \emph{Instrument} your program so that when a reporting flag is
  set it prints the number of iterations/recursions. Use your
  instrumentation to check the assertion in Problem~\ref{logtime}.

\item Submit hard copy of your program. If possible, do that in this
\LaTeX{} document using the \verb!listings! package. This is a
particularly useful part of the homework for cs students.

\end{itemize}

\item
Improve your solution to the previous problem so that your program
both finds the gcd of its input values and also finds the coefficients
for a linear integral combination of the inputs that produces the
gcd.

I'd still rather your function do no printing, but that's harder to
arrange now that there are three integer outputs rather than just
one. Do that if you can, but if you can't don't worry.

\item If you can, write both programs so that they run in constant
  space. In particular, no recursive calls, since that would create a
  logarithmic number of stack frames.

This is easier for the first program than the second.

\end{enumerate}

\item Calculate
%
\begin{equation*}
54^{100} \mm{101}
\end{equation*}
%
using the fast
\textbf{Right-to-left binary method} described at
\url{http://en.wikipedia.org/wiki/Modular_exponentiation}.

Use a calculator along the way.

If you don't like the wikipedia discussion you can find lots of others
by googling \verb!fast modular exponentiation!.
```

```
This is the computation I botched at the end of lecture.

Note: you should get $1$ as the answer -- that's Fermat's Little
Theorem.

\item Large primes.

Note: you can look up the answers to almost all these questions. In
fact I've asked you to do that for the last few. I'd rather you didn't
at the beginning -- you'll learn more that way.

\begin{enumerate}
\item Prove

\begin{theorem}\label{thm:mersenne}
If $2^n -1$ is prime then $n$ is prime.
\end{theorem}

Hint: If $n = ab$ is not prime then $2^n = (2^a)^b$. Then use a finite
geometric series.

Primes of this form are called ``Mersenne primes''. The first few are
$3 = 2^2-1$, $7=2^3-1$, $31=2^5-1$ and $127=2^7-1$.

\item State the \emph{converse} of
Theorem~\ref{thm:mersenne}. Then show that it is false.

Hint. Try to continue the list above in the obvious way.

\item Look up some information about how the largest known   prime has
  increased over time. Argue from the data that the  logarithm of the
  logarithm of the largest known prime is growing. That means the
  largest known prime is growing much faster than exponentially!

\item Check Chris Caldwell's Prime Pages at
  \url{http://primes.utm.edu/}. Read about GIMPS at
  \url{http://www.mersenne.org/}. Tell me something you found
  particularly interesting (not just something from the first page).

\end{enumerate}

\item Show that $2^{340} \equiv 1 \mod{341}$ even though $341$ is not
  prime. What is the connection between this result and the converse
  of Fermat's Little Theorem?

\item Eratosthenes' RSA public key is
%
\begin{align*}
n & = 10967535067 \\
e & = 1051
\end{align*}

Archimedes encrypts his message with this key and sends Eratosthenes
%
\begin{equation*}
C = 1963501580
\end{equation*}

\begin{enumerate}
```

```
\item Break the encryption. (Use any tools you like; tell me how you
  did it.)

\item Why is the message appropriate?

\item What is each of these Greeks famous for?
\end{enumerate}

\end{enumerate}
\newpage

\verbatiminput{\jobname}

\end{document}
```