MATH 895 Assignment 1, Summer 2011

Instructor: Michael Monagan

Please hand in the assignment by 3:30pm Monday May 27th.
Late Penalty -20% off for up to one day late. Zero after that.
For Maple problems, please submit a printout of a Maple worksheet containing Maple code
and the execution of examples.

References: Sections 4.5–4.9 of Geddes, Czapor and Labahn and/or sections 8.2,8.3,9.1 of
von zur Gathen and Gerhard.

**Question 1 Implementing the FFT.**

Implement the FFT, the forward transform (Algorithm 4.4), in Maple. Program it to take
as input a Maple list of integers, e.g., \([a_0, a_1, ..., a_{m-1}, 0, ..., 0]\) for \(a(x)\), and to output a list
of integers. To make your implementation efficient optimize it for the case \(n = 2\).

Check that your implementation is correct by computing the Fourier transform of the
following polynomial \(f(x)\) using the prime \(p = 7 \times 2^{20} + 1\), then applying the inverse FFT
to get back to \(f(x)\). You will need a primitive \(n = 64\)'th root of unity.

```maple
> p := 7*2^20+1;
> f := Randpoly(50,x) mod p;
> a := [seq(coeff(f,x,i),i=0..50), 0$13];
```

Time your implementation on inputs of suitable degree \(d\) and check that the complexity
of your implementation is \(O(d \log d)\) and NOT \(O(d^2)\).

**Question 2 Integer multiplication using the FFT.**

Design and implement an algorithm which uses the FFT to multiply two large integers \(a\) and
\(b\) using three machine primes \(p_1, p_2, p_3\) and then applying the Chinese remainder theorem.
Do this using the base \(B = 2^{30}\) and the primes \(p_1 = 2^{24} \times 10 + 1\), \(p_2 = 2^{24} \times 28 + 1\) and
\(p_3 = 2^{24} \times 45 + 1\). Test your algorithm on multiplying \(a \times b\) where

```maple
> r := rand(2^(-10^-6)):
> a := r():
> b := r():
```

To split up a large integer \(A\) into blocks base \(B\) use the Maple command \(\text{convert}(A, \text{base}, B)\)
Question 3 Shönhage Strassen integer multiplication.

Implement the Schönhage-Strassen integer multiplication algorithm in Maple. It uses a large integer modulus of the form \( p = 2^{2^r} + 1 \). To divide by \( p \) just use Maple so that you can reuse your FFT code from question 2 here. Also, use Maple for doing the multiplications in \( C := [A_i \times B_i \mod p, \text{ for } i = 1..n] \), i.e. don’t make these multiplications recursive. Test your algorithm on the example in question 3.

Question 4 Fast Division

Consider dividing \( a \) by \( b \) in \( F[x] \) where \( \text{deg } a = d, \text{deg } b = m \) for \( d \geq m \geq 0 \). Program the Newton iteration (Algorithm 4.6) recursively in Maple for \( F = \mathbb{Z}_p \) to compute \( q_r = a_r / b_r \) by computing \( \frac{1}{b_r} \) as a power series to \( O(x^n) \) where \( n = d - m + 1 \).

To make the algorithm work efficiently when \( n \) is not a power of 2, compute \( y = \frac{1}{b_r} \) recursively to order \( O(x^{\lceil n/2 \rceil}) \).

To reduce a polynomial \( f \) modulo \( x^n \) you could use the Maple command \( \text{rem}(f, x^n, x) \). Use instead the following Maple command \( \text{convert(taylor(f, x, n), polynom)} \).

Test your algorithm on the following problem in \( \mathbb{Z}_p[x] \).

\[
\text{Let } M(n) \text{ be the cost of multiplying two polynomials in } F[x] \text{ of degree } n. \text{ Let } T(n) \text{ be the cost of computing } \frac{1}{b_r} \text{ to order } O(x^n). \text{ In class we said that the cost of Newton’s method for } n = 2^k \text{ is }
\]

\[
T(n) \leq T(n/2) + M(n) + M(n/2) + cn
\]

for \( n > 1 \) and \( T(1) = d \) for some constants \( c > 0 \) and \( d > 0 \). Show that

\[
T(n) \leq 3M(n) + 2cn + d
\]

and conclude that computing \( \frac{1}{b_r} \) to order \( O(x^n) \) costs asymptotically 3 polynomial multiplications of degree \( n \).