MATH 497, MATH 895, CMPT 894.
Assignment 3, Summer 2007

Instructor: Michael Monagan

Please hand in the assignment by 2:30pm on July 4th before class starts.
Late Penalty -20% off for each day late.
Please submit a printout of a Maple worksheet containing Maple code and output.
Use any tools from the Maple library, e.g. content(...), Content(...) mod p, divide(...), Divide(...) mod p, eval(...) mod p, Interp(...) mod p, Linsolve(A,b) mod p, chrem(...), etc.

Brown's dense modular GCD algorithm for $\mathbb{Z}[x_1, x_2, ...x_n]$

REFERENCE: Section 7.4 of the Geddes text.

(a) Let $a, b \in \mathbb{Z}[x_1, x_2, ..., x_n]$. Let $g = \gcd(a, b)$, $\bar{a} = a/g$ and $\bar{b} = b/g$. For the modular GCD algorithm in $\mathbb{Z}[x]$ (one variable) we said a prime p is *bad* if $p|\mathrm{lc}_x a$ and a prime p is *unlucky* if $\deg_x \gcd(\bar{a} \bmod p, \bar{b} \bmod p)) > 0$ and we apply Lemma 7.3 (see text) to identify the unlucky primes.

For $a, b, \in \mathbb{Z}[x_1, x_2, ..., x_n]$ we need to generalize these definitions and also define bad evaluation points and unlucky evaluation points. Consider the following two polynomials in $\mathbb{Z}[x, y, z]$

$$a = (yx + yz - 1)(x + 7y(z^2 - 1) + 1), \quad b = (yx + yz - 1)(x + 7y(z^3 - 1) + 1).$$

Assuming we first evaluate z then y, thus reducing to univariate gcd computations in $\mathbb{Z}_p[x]$ for some prime p, identify all bad primes, unlucky primes, bad evaluation points, and unlucky evaluation points. Now explain how your algorithm detects unlucky primes and unlucky evaluation points.

Let x_1 be the main variable. We will say a prime p *introduces an unlucky content* if $\deg(c \in \mathbb{Z}_p[x_2, ..., x_n]) > 0$ where $c = \gcd(\mathrm{cont}_{x_1}(\bar{a} \bmod p), \mathrm{cont}_{x_1}(\bar{b} \bmod p)))$. Analogously we define evaluation points which introduce an unlucky content. Consider

$$a = (x + yz + 1)(yx + y + 7z), \quad b = (x + yz + 1)(yx^2 + 7zx + y).$$

If x is the main variable, identify which primes introduce an unlucky content and which evaluation points for z introduce an unlucky content. Now explain how your algorithm detects primes and evaluation points that introduce an unlucky content.

(b) Implement the modular GCD algorithm of section 7.4 in Maple. Implement two subroutines, subroutine MGCD that computes the GCD modulo a sequence of primes (use 4 digit primes), and subroutine PGCD that computes the GCD at a sequence of evaluation points (use $0, 1, 2, ...$ for the evaluation points). Note, subroutine PGCD is recursive. Test your algorithm on the following examples polynomials in $\mathbb{Z}[x, y, z]$. Use x as the main variable. First evaluate out z then y.

```
> c := x^3+y^3+z^3+1; d := x^3-y^3-z^3+1;
> g := x^4-123454321*y*z^2*x^2+1;
> MGCD(c,d,[x,y,z]);
> MGCD(expand(g*c),expand(g*d),[x,y,z]);
> MGCD(expand(g^2*c),expand(g^2*d),[x,y,z]);

> g := z*y*x^3+1; c := (z-1)*x+y+1; d := (z^2-1)*x+y+1;
> MGCD(expand(g*c),expand(g*d),[x,y,z]);
> g := x^4+z^2*y^2*x^2+1; c := x^4+z*y*x^2+1; d := x^4+1;
> MGCD(expand(g*c),expand(g*d),[x,y,z]);
> g := x^4+z^2*y^2*x^2+1; c := z*x^4+z*x^2+y; d := z*x^4+z^2*x^2+y;
> MGCD(expand(g*c),expand(g*d),[x,y,z]);
```

Please make your MGCD procedure print out the sequence of primes it uses using `printf(" p=%d\n",p);` .

Please make your PGCD procedure print out the sequence of evaluation points α that it uses for each variable u using `printf("  %a=%d\n",u,alpha);`

Zippel's sparse modular GCD algorithm for $\mathbb{Z}[x_1, ..., x_n]$

Graduate students only.

REFERENCES: Section 7.5 of the Geddes text and the paper "Algorithms for the Non-monic case of the Sparse Modular GCD algorithm" by de Kleine, Monagan and Wittkopf.

Modify subroutine PGCD to use sparse interpolation. You may assume that the gcd g is monic. You may modify subroutine MGCD to also use sparse interpolation if you wish.

Run both your sparse algorithm and dense algorithm on this input. Count the number of univariate gcd computations in $\mathbb{Z}_p[z]$ that each algorithm does.

```
> g := 2*x^8 + (u^8*v - 3*v^8*y + y^8*u)*x^4 + (w^8*z - 3*z^8*w + 1);
> c := 4*x^8 + 5*w^4*x^4 + 2*y^4*z^4 + 3*u^4*v^4 + 1;
> d := 6*x^8 - 5*y^4*x^4 - 4*u^4*v^4 - 3*w^4*z^4 - 2;
> a := expand(g*c):
> b := expand(g*d):
> MGCD(a,b,[x,u,v,w,y,z]);
```