Numerical Algorithms

Michiel Baird

27 February 2009

1. Euclid's Alogrithm

```
int GCD(int a, int b){
        if (b == 0)
                return a;
        else
                return GCD(b,a%b);
}
```

2. Sieve of Eratosthenes

```
vector <int> Sieve(int n){
        bool * ptest;
        vector <int> primes;
        ptest = new bool[n+1];
        for (int i = 2;i <= n;i++)
                ptest[i] = true;
        for (int i = 2; i <= n;i++)
                if ptest[i]{
                        primes.push_back(i);
                        for (int j = i*i;j <=N;j+=i)
                            ptest[j] = false;
                }
        delete [] primes;
        return primes;
}
```

3. Horner's Rule

```
int horner(int x,vector <int> A){
        int Answer = A[0];
        for (int i = 1;i < A.size();i++){
                Answer  *= X;
                Answer += A[i];
        }
        return Answer;
```

```
}
```

4. Factoring

```
vector <int> factoring(int n){
        vector <int> factors;
        int Ans = n;
        for (int i = 1;i<=N;i++){
                while (Ans % i == 0){
                        factors.push_back(i);
                        Ans /= i;
                }
                if (Ans == 1) break;
        }
        return factors;
    }
```

5. Efficient Exponentiation(Left to Right)

```
int LeftToRight(int a, int b){
        if (b == 0)
                return 1;
        int c =  LeftToRight(a,b / 2);
        if (b %2 )
                return a*c*c;
        else
                return c*c;
        }
}
```

6. Efficient Exponentiation(Right to Left)

```
int RightToLeft(int a, int b){
        int index = a;
        int Answer = 1;
        while(b){
                if (b % 2)
                        Answer  *= Index;
                b /= 2;
                Index *= Index;
        }
        return Answer;
}
```