Lecture 6

Debugging, profiling, and more on coding

Homework Review

1) Get filename and row/column from user
2) Read file into array
3) Transpose result to correct ordering
4) Plot

Homework Review

1) Get filename and row/column from user

```
from sys import argv,exit
from pylab import *

try: filename, xcol, ycol=argv[1], int(argv[2]),
int(argv[3])
except:
 print "Usage: %s <filename> <X col> <Y col>"%argv[0]
 sys.exit(1)
```

Homework Review

2) Read file into array & 3) transpose

```
fin=file(filename,"r")

# make a list of lists. Each element in the outer list is
one row of values
ary=[]
for line in fin:
    ary.append([])
    for val in line.split(","):
        ary[-1].append(float(val))

# now "transpose" the list, so each list in
# the outer list is a column instead of a row
ary2=[ []*len(ary[0]) ]
for row in ary:
    for col,val in enumerate(row):
        ary2[col].append(val)
```

Let’s try it !

Debugging

- print statements ?
- traceback module: print_exc()

Debuggers

- Built in IDLE (broken on Mac 10.6 system python)
- http://wiki.python.org/moin/PythonDebuggers
- http://www-pcmdi.llnl.gov/svn/repository/cdat/trunk/Packages/pydebug/Lib/pydebug.py

Homework Review

2) Read file into array

```
fin=file(filename,"r")
ary=array([[float(i) for i in line.split(",")] for line in
fin])

3) Transpose result to correct ordering
ary=ary.transpose()

4) Plot
plot(ary[xcol],ary[ycol])
show()
```

Homework Review

```
from sys import argv,exit
from pylab import *

try: filename, xcol, ycol=argv[1], int(argv[2]),
int(argv[3])
except:
 print "Usage: %s <filename> <X col> <Y col>"%argv[0]
 sys.exit(1)

fin=file(filename,"r")
ary=array([[float(i) for i in line[:-1].split(“,”)] for
line in fin]).transpose()

plot(ary[xcol],ary[ycol])
show()
```

Homework Review

```
from sys import argv,exit
from pylab import *

def safe_float(x):
   try: return float(x)
   except: return(0.0)

try: filename, xcol, ycol=argv[1], int(argv[2]), int(argv[3])
except:
   print "Usage: %s <filename> <X col> <Y col>"%argv[0]
   sys.exit(1)

fin=file(filename,"r")
ary=array([[safe_float(i) for i in line.split(",")] \
                         for line in fin]).transpose()

plot(ary[xcol],ary[ycol])
show()
```

Homework Review

```
from sys import argv,exit
from pylab import *

def safe_float(x):
    try: return float(x)
    except: return(0.0)

try: filename, xcol, ycol=argv[1], int(argv[2]), int(argv[3])
except:
    print "Usage: %s <filename> <X col> <Y col>"%argv[0]
    sys.exit(1)

ary=genfromtxt(filename,delimiter=',',skip_header=1)
ary=ary.transpose()

plot(ary[xcol],ary[ycol])
show()
```

Homework Review

- CSV module ?

Find the first n primes

- Check to see if each number is divisible by every other number

```
for i in range(40000):
    for j in range(2,i/2):
        if i%j==0 : break
    else: print i

7.65
```

Find the first n primes

- What if we skip even numbers, which are divisible by 2, as well as even factors, which cannot be prime

```
for i in range(3,40000,2):
    for j in range(3,i/2,2):
        if i%j==0 : break
    else: print i
```

3.0

Find the first n primes

- Actually, we just need to find the FIRST prime factor to show that it isn't prime. The first prime factor, if it exists, is guaranteed to be <sqrt(number). Think about it...

```
for i in range(3,40000,2):
    for j in range(3,int(i**.5)+1,2):
        if i%j==0 : break
    else: print i
```

.084

Find the first n primes

- The only factors we need to check are themselves prime numbers.

```
primes=[3]
for i in range(5,10000,2):
    for j in primes:
        if i%j==0: break
    else: primes.append(i)

for i in primes: print i

1.0
```

Find the first n primes

```
primes=[3]
m=1
for i in range(5,10000,2):
    while primes[m-1]**2<i : m+=1
    for j in primes[:m]:
        if i%j==0: break
    else: primes.append(i)
.049
```

Find the first n primes

- A 1-line program which does the same thing. Not the fastest, and certainly not easy to follow.

```
[i for i in range(3,10000,2) if len([j for j in
                    range(3,int(i**.5)+1,2) if i%j==0])==0]
```

.196

Sieve of Eratosthenes

- remove non-primes

```
primes=set(range(40001))
for i in xrange(2,201):
    bad=range(i*2,40001,i)
    primes.difference_update(bad)

print primes,len(primes)

0.022
```

Profiling

- 3 built-in profilers
  - profile, cProfile, hotshot
  - python -mcProfile script.py
  - line_profiler
- ‘C’ level profilers
  - valgrind/cachegrind (linux)
  - dtrace (mac)

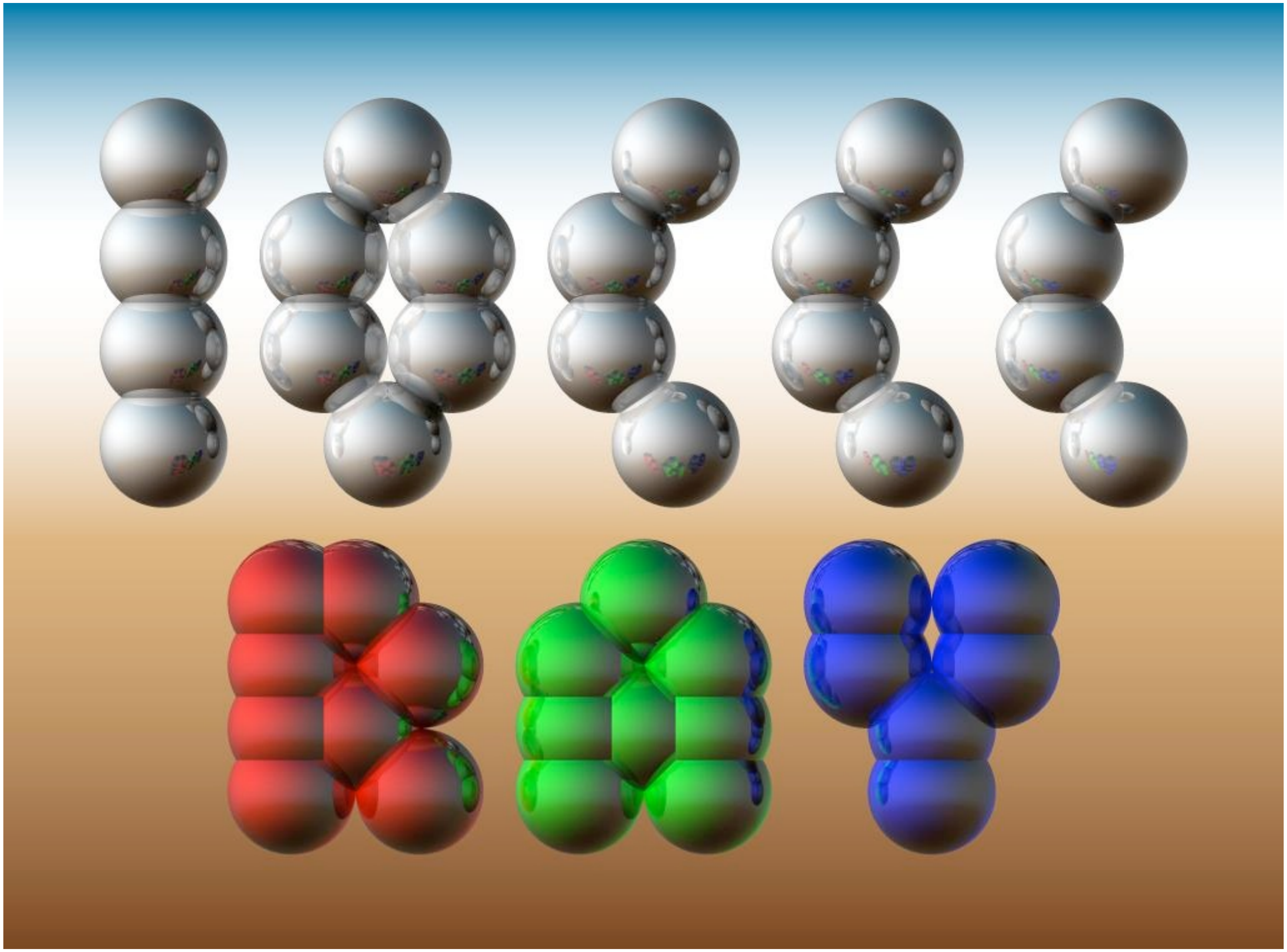
Obfuscated C Contest

Goal of the competition is to produce a C program which does something interesting but is as difficult to read as possible, and may be internally complicated. This is much like the poetry of programming. Here is an example of a recent winner:

Obfuscated C Contest

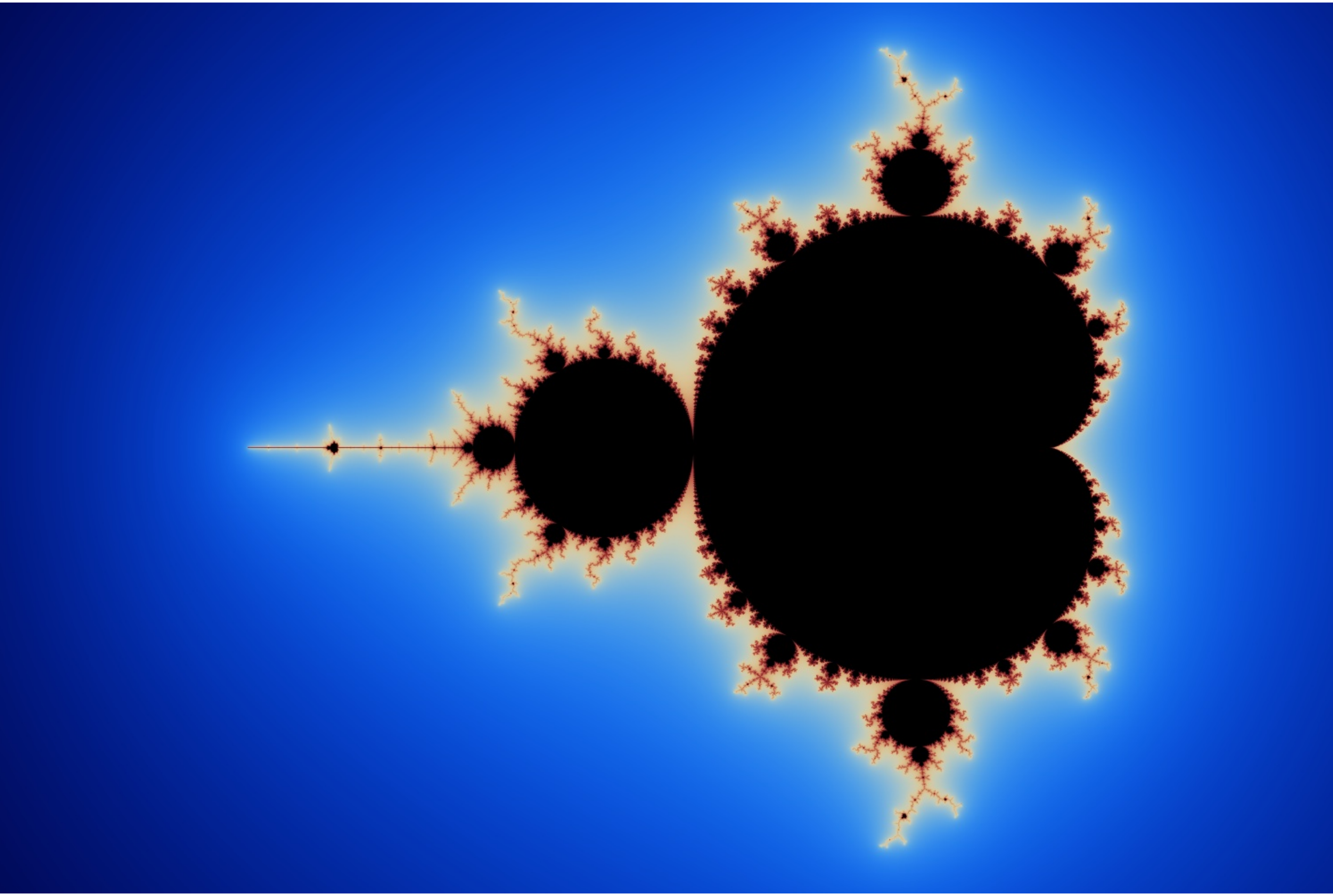
```
                              X=1024; Y=768; A=3;

J=0;K=-10;L=-7;M=1296;N=36;O=255;P=9;_=1<<15;E;S;C;D;F(b){E="1""111886:6:??AAF"
"FHHMMOO55557799@@>>>BBBGGIIKK"[b]-64;C="C@=::C@@==@=:C@=:C@=:C5""31/513/5131/"
"31/531/53"[b ]-64;S=b<22?9:0;D=2;}I(x,Y,X){Y?(X^=Y,X*X>x?(X^=Y):0,  I (x,Y/2,X
)):(E=X);      }H(x){I(x,    _,0);}p;q(        c,x,y,z,k,l,m,a,          b){F(c
);x-=E*M     ;y-=S*M           ;z-=C*M         ;b=x*       x/M+         y*y/M+z
*z/M-D*D    *M;a=-x              *k/M     -y*l/M-z        *m/M;    p=((b=a*a/M-
b)>=0?(I    (b*M,_      ,0),b    =E,      a+(a>b      ?-b:b)):     -1.0);}Z;W;o
(c,x,y,     z,k,l,    m,a){Z=!    c?      -1:Z;c     <44?(q(c,x         ,y,z,k,
l,m,0,0     ),(p>      0&&c!=     a&&        (p<W         ||Z<0)          )?(W=
p,Z=c):     0,o(c+         1,    x,y,z,        k,l,          m,a)):0     ;}Q;T;
U;u;v;w    ;n(e,f,g,            h,i,j,d,a,    b,V){o(0      ,e,f,g,h,i,j,a);d>0
&&Z>=0? (e+=h*W/M,f+=i*W/M,g+=j*W/M,F(Z),u=e-E*M,v=f-S*M,w=g-C*M,b=(-2*u-2*v+w)
/3,H(u*u+v*v+w*w),b/=D,b*=b,b*=200,b/=(M*M),V=Z,E!=0?(u=-u*M/E,v=-v*M/E,w=-w*M/
E):0,E=(h*u+i*v+j*w)/M,h-=u*E/(M/2),i-=v*E/(M/2),j-=w*E/(M/2),n(e,f,g,h,i,j,d-1
,Z,0,0),Q/=2,T/=2,       U/=2,V=V<22?7:  (V<30?1:(V<38?2:(V<44?4:(V==44?6:3)))))
,Q+=V&1?b:0,T                +=V&2?b        :0,U+=V    &4?b:0)     :(d==P?(g+=2
,j=g>0?g/8:g/     20):0,j    >0?(U=     j    *j/M,Q      =255-    250*U/M,T=255
-150*U/M,U=255    -100    *U/M):(U    =j*j     /M,U<M           /5?(Q=255-210*U
/M,T=255-435*U           /M,U=255    -720*      U/M):(U       -=M/5,Q=213-110*U
/M,T=168-113*U    /       M,U=111               -85*U/M)      ),d!=P?(Q/=2,T/=2
,U/=2):0);Q=Q<    0?0:      Q>O?     O:          Q;T=T<0?    0:T>O?O:T;U=U<0?0:
U>O?O:U;}R;G;B    ;t(x,y     ,a,    b){n(M*J+M    *40*(A*x   +a)/X/A-M*20,M*K,M
*L-M*30*(A*y+b)/Y/A+M*15,0,M,0,P,  -1,0,0);R+=Q    ;G+=T;B   +=U;++a<A?t(x,y,a,
b):(++b<A?t(x,y,0,b):0);}r(x,y){R=G=B=0;t(x,y,0,0);x<X?(printf("%c%c%c",R/A/A,G
/A/A,B/A/A),r(x+1,y)):0;}s(y){r(0,--y?s(y),y:y);}main(){printf("P6\n%i %i\n255"
                                "\n",X,Y);s(Y);}
```

Obfuscated C Contest



Obfuscated Python ?

```
_                                      =   (
                                        255,
                                      lambda
                               V       ,B,c
                             :c   and Y(V*V+B,B,  c
                               -1)if(abs(V)<6)else
               (              2+c-4*abs(V)**-0.4)/i
                 )  ;v,      x=1500,1000;C=range(v*x
                  );import  struct;P=struct.pack;M,\
            j  ='<QIIHHHH',open('M.bmp','wb').write
for X in j('BM'+P(M,v*x*3+26,26,12,v,x,1,24))or C:
            i  ,Y=_;j(P('BBB',*(lambda T:(T*80+T**9
                  *i-950*T  **99,T*70-880*T**18+701*
                 T  **9     ,T*i**(1-T**45*2)))(sum(
               [              Y(0,(A%3/3.+X%v+(X/v+
                               A/3/3.-x/2)/1j)*2.5
                             /x   -2.7,i)**2 for  \
                               A       in C
                                      [:9]])
                                        /9)
                                       )   )
```



Homework 6

1. Start with your CSV plotting program. You will now modify this program so instead of plotting X vs Y values, it computes a histogram of one user specified column from a CSV file, and plots it as a bar chart using MatPlotlib. The user should have the option of specifying the minimum, maximum and/or number of bins for the histogram. Any value the user doesn't specify should be selected in some sensible way automatically by your program. You may get the values from the user with argv[], using raw_input, or any other method you like. Several sample files are available to test your program with. Make sure your program does something reasonable with all of the examples. You need only turn in the code, not the output.