Non Linear Optimisation I Assignment 3

Exercise 3.3

```
function [B, flag] = modNewton(H,beta)

flag = 0;
if (norm(H-H','fro')~=0)
	fprintf('The given hessian is not symmetric: error \n')
	return;
end

if (beta<=1)
	fprintf(' The value of beta needs to be greater than 1: error \n')
	return;
end


if (norm(H,'fro')==0)
	epsi = 1;
else
	epsi = norm(H,2)/beta;
	[V,D] = eig(H);
end

D_bar = max(D,epsi*eye(size(H)));

if norm(D_bar-D,'fro') ~=0
	flag =1;
	B = V*D_bar*V';
else
	B = H;
end

end
```

Output:

```
B =

	1	0
	0	-2

modNewton(B,10)
```

ans =

```
    1.0000         0
    0         0.2000
```

Exercise 3.4

a)
Function 1:

```
function [x,F,G,H,iter,status] = uncMIN(fun,x0,step,maxit,printlevel,tol)

x = x0;
count = 1;
status =0;
fh = str2func(fun);
tau = 0.5;
nu = 0.7;
alpha_init =1;

for i = 1:maxit

    [F_iter,G_iter,H_iter] = fh(x);
    alpha_iter= alpha_init;

    if (i==1)
    F_x0 = F_iter;
    end

    if((norm(F_iter,'fro')/norm(F_x0,'fro')) < tol)
    status =1;
    break;
    end

    if (count < printlevel)
    fprintf('\n');
    fprintf('Iteration %d || Function Value : %f \n', i,norm(F_iter,'fro'));
    end



    if (step == 0)
    p_iter = -G_iter;
    alpha_iter =0.01;
```

```
        else
        [B_iter,~] = modNewton(H_iter,10);
        p_iter = -B_iter\G_iter;

        for l = 1:100

        [F_iter_l,~,~] = fh(x+ alpha_iter*p_iter);

        if (F_iter_l > F_iter+nu*alpha_iter*G_iter'*p_iter)
        alpha_iter = alpha_iter*tau;
        end

        end
        end

        x = x + alpha_iter*p_iter;
        count = count + 1;

end

iter = i;
F = F_iter;
G= G_iter;
H= H_iter;

end
```

b)

Function 2 :

```
function [F,G,H] = fun(x)
syms y1 y2
my_func = 10*(y2-y1^2)^2 +(y1-1)^2;

grad = gradient(my_func,[y1;y2]);
hess = hessian(my_func,[y1;y2]);
F = double(subs(my_func,[y1;y2],x));
G = double(subs(grad,[y1;y2],x));
H = double(subs(hess,[y1;y2],x));

end
```

<u>Results:</u>

Steepest Descent:

```
[x,F,G,H,iter,status] = uncMIN('fun',[0,0]',0,1000,15,10e-04)

Iteration 1 || Function Value : 1.000000

Iteration 2 || Function Value : 0.960402

Iteration 3 || Function Value : 0.922396

Iteration 4 || Function Value : 0.885987

Iteration 5 || Function Value : 0.851197

Iteration 6 || Function Value : 0.818045

Iteration 7 || Function Value : 0.786541

Iteration 8 || Function Value : 0.756676

Iteration 9 || Function Value : 0.728419

Iteration 10 || Function Value : 0.701726

Iteration 11 || Function Value : 0.676534

Iteration 12 || Function Value : 0.652770

Iteration 13 || Function Value : 0.630354

Iteration 14 || Function Value : 0.609203

x =

        0.9687
        0.9371


F =

  9.9662e-04


G =

  -0.0130
  -0.0256
```

H =

```
   77.1204  -38.7477
  -38.7477   20.0000
```

iter =

```
   588
```

status =

```
           1
```

Modified Newton:

[x,F,G,H,iter,status] = uncMIN('fun',[0,0]',1,1000,15,10e-04)

Output:

Iteration 1 || Function Value : 1.000000

Iteration 2 || Function Value : 0.601562

Iteration 3 || Function Value : 0.375121

Iteration 4 || Function Value : 0.182313

Iteration 5 || Function Value : 0.125564

Iteration 6 || Function Value : 0.095516

Iteration 7 || Function Value : 0.075858

Iteration 8 || Function Value : 0.061898

Iteration 9 || Function Value : 0.051471

Iteration 10 || Function Value : 0.043407

Iteration 11 || Function Value : 0.037010

Iteration 12 || Function Value : 0.031834

Iteration 13 || Function Value : 0.027580

Iteration 14 || Function Value : 0.024041

```
x =

    0.9697
    0.9390


F =

  9.3552e-04


G =

  -0.0125
  -0.0249


H =

   77.2701  -38.7867
  -38.7867   20.0000


iter =

    46


status =

     1
```