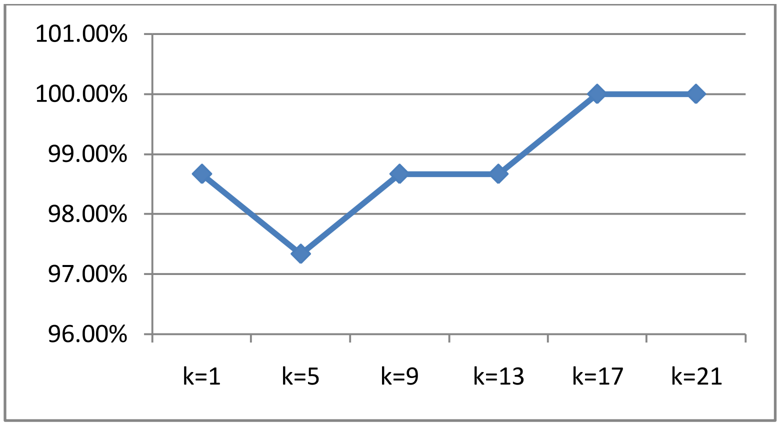
Question 2:

a) The k values and the related confusion matrices are given below.

K Values	K=1	K=5	K=9	K=13	K=17	K=21
Confusion Matrices	25 0 0 0 24 1 0 0 25	25 0 0 0 23 2 0 0 25	25 0 0 0 24 1 0 0 25	25 0 0 0 24 1 0 0 25	25 0 0 0 25 0 0 0 25	25 0 0 0 25 0 0 0 25

b) The accuracy vs. k value graph is given below



c) As we can see from the accuracy graph above, all k values yield a high accuracy. K values of 1, 9 and 13 get the same accuracy values and there is a decrease in the accuracy value for k=5. For k=17 and k=21 we achieve perfect accuracy, i.e. we were able to correctly classify all the instances.

The MATLAB code that I have used for the above solution is as follows:

```
ks = [1 5 9 13 17 21]';

load iris.dat;
counter = 1;
iris1train = iris(counter:(counter+24),:);
counter = counter+25;
iris1test = iris(counter:(counter+24),:);
counter = counter+25;

iris2train = iris(counter:(counter+24),:);
counter = counter+25;
iris2test = iris(counter:(counter+24),:);
counter = counter+25;

iris3train = iris(counter:(counter+24),:);
counter = counter+25;
iris3test = iris(counter:(counter+24),:);
counter = counter+25;

trainSet = [iris1train;iris2train;iris3train];

for i=1:size(ks,1)
    confusionMatrix = zeros(3,3);
    kValue = ks(i);
    for j=1:size(iris1test,1)
        myClass = myKnn(iris1test(j,:),trainSet, kValue);
        confusionMatrix(1,myClass) = confusionMatrix(1,myClass) + 1;
    end
    for k=1:size(iris2test,1)
        myClass = myKnn(iris2test(k,:),trainSet, kValue);
        confusionMatrix(2,myClass) = confusionMatrix(2,myClass) + 1;
    end
    for l=1:size(iris3test,1)
        myClass = myKnn(iris3test(k,:),trainSet, kValue);
        confusionMatrix(3,myClass) = confusionMatrix(3,myClass) + 1;
    end
    kValue
    confusionMatrix
end

function [myClass] = myKnn(instance, trainSet, kValue)
classWins = zeros(3,1);

for i = 1:kValue
   closestNeighbor = trainSet(1,:);
   minDist = sqrt(sum((instance(1,(1:(size(instance,2)-1))) - trainSet(1,(1:(size(trainSet,2)-1)))).^2));
   myIndex = 1;
   for j = 2:size(trainSet,1)
      tempDist =  sqrt(sum((instance(1,(1:(size(instance,2)-1))) - trainSet(j,(1:(size(trainSet,2)-1)))).^2));
      if tempDist < minDist
         minDist = tempDist;
         closestNeighbor = trainSet(j,:);
         myIndex = j;
      end
   end
   trainSet(myIndex,:) = [];
   classWins(closestNeighbor(1,size(closestNeighbor,2))) = classWins(closestNeighbor(1,size(closestNeighbor,2))) + 1;
end

[C myClass] = max(classWins);

end
```