1

A general graph has been used to make the program work in case of descendants having more than 1 top ancestor.Binary tree cannot be used since a person might have more than two children.A 2-3 tree also won't work for the same reason above.
Hence we used a general graph or in this case tree since it is given in the question that people in a geneology never have children with each other, but only with others outside the geneology. Here 'Sarah' stands out as she has neither descendants nor ancestors .Hence only one tree exists starting from 'James'. In the code , I have considered it as more like a general graph starting from ancestors to leaf. Original ancestors are identified by the fact that they don't have parents.

2

Edge information is stored as two dimensional array.Each row denotes ith individual and each column contains the age information at the birth of jth child . The child index per individual has been stored in a separate 2-D array c[i][j] where i=individual and j=jth child index in the main array(nodes).
The cno[i](ith element of array cno)contains the no. of children of the ith node.The algorithm is based on recursion. The no. of descendants of each individual is traced down to the bottom most node(which has no children). Hence no. of descendants have been calculated.The problem which had occurred with the DP approach i had tried out was that the hierarchical level of the nodes couldn't be determined. Hence recursion was implemented instead of DP. Since recursion was used, time complexity is exponential O(E)
In the worst case performance, it is of the order of

$$timecomplexity = O(b^k)$$

where b=branching factor and k=depth.
We cannot accurately determine time complexity since the no. of branches per node is random and hence it cannot be calculated by known methods.

3

Here again recursion can be implemented.The graph has been traversed depth-wise (DFS). A counter was used to tell whether it is the great grand child (or if they exist) and the names are printed.This was done for every node.The time complexity as mentioned before cannot be accurately determined since the number of branches of each node is unknown.But in general

$$timecomplexity = O(nb^k)$$

where b=*branching factor* and k=*no. of generations.*[k=*depth* in worst case.] Here in worst case

$$timecomplexity = O(nb^4)$$

We cannot accurately determine time complexity since the no. of branches per node is random and hence it cannot be calculated by known methods.