The basic control procedure for a ray tracer consists of a simple recursive procedure that reflects the action at a node where, in general, two rays are spawned. Thus the procedure will contain two calls to itself, one for the transmitted and one for the reflected ray. We can summarize the action as:

ShootRay (*ray structure*)
    intersection test
    if *ray intersects an object*
        get normal at intersection point
        calculate local intensity (I_{local})
        decrement current depth of trace
        if *depth of trace* > 0
            calculate and shoot the reflected ray
            calculate and shoot the refracted ray

where the last two lines imply a recursive call of ShootRay(). This is the basic control procedure. Around the recursive calls there has to be some more detail which is:

Calculate and shoot reflected ray *elaborates as*
    if *object is a reflecting object*
        calculate reflection vector and include in the ray structure
        Ray Origin := intersection point
        Attenuate the ray (multiply the current k_{rg} *by its value at the previous invocation)*
        ShootRay(*reflected ray structure*)
        if *reflected ray intersects an object*
            combine colours (k_{rg} *I*) *with* I_{local}

Calculate and shoot refracted ray *elaborates as*
    if *object is a refracting object*
        if *ray is entering object*
            accumulate refractive index
            increment number of objects that the ray is currently inside
            calculate refraction vector and include in refracted ray structure
        else
            de-accumulate refractive index
            decrement number of objects that the ray is currently inside
            calculate refraction vector and include in refracted ray structure
        Ray origin := intersection point
        Attenuate ray (k_{tg})
        if *refracted ray intersects an object*
            combine colours (k_{tg} *I*) *with* I_{local}