Loop Over All Leaf Nodes and Print `id` Field

```
// After last loop, i is first_leaf.
for ( ; p->n_nodes > i; i++) {
    fprintf (out, "%d\n",
             p->node[i].id);
}
return (0 == fclose (out));
```

Loop over all leaf nodes.

Print `id` field.

Close file and return 0 or 1.

In Binary Version, First Open the File as a Stream

What about the binary version?

```
int32_t write_pyr_tree_binary
    (pyr_tree_t* p, const char* fname)
{
    FILE* out;
    if (NULL == (out =
                 fopen (fname, "w"))) {
        return 0;
    }
```

First part is identical to the ASCII version.

Write Number of Nodes Followed by Node Array

```
int32_t rval =
    (1 == fwrite (&p->n_nodes,
     sizeof (p->n_nodes), 1, out) &&
     p->n_nodes == fwrite
      (p->node, sizeof (p->node[0]),
       p->n_nodes, out));
fclose (out);
return rval;
```

Write `n_nodes` to output file.

Write node array to output file.

Close Output Stream and Return Success or Failure

```
int32_t rval =
    (1 == fwrite (&p->n_nodes,
     sizeof (p->n_nodes), 1, out) &&
     p->n_nodes == fwrite
      (p->node, sizeof (p->node[0]),
       p->n_nodes, out));
fclose (out);
return rval;
```

Return success if both writes succeed.

Close the output stream.

Return success or failure.

