En el código dado, se está liberando solo la raíz del árbol mediante la función free(root) en la función arbolBinario. Esto no libera toda la memoria que se asignó para crear los nodos en el árbol. Para evitar los memory leaks, es necesario liberar la memoria de cada nodo creado en el árbol.

Para hacer esto, se puede recorrer el árbol en postorden (primero visitando los hijos, luego el nodo actual) y liberar la memoria de cada nodo visitado. En lugar de llamar free(root), se debe llamar a una función liberarArbol que recorra el árbol y libere la memoria de cada nodo.

La función liberarArbol podría ser algo así:

```
scss
Copy code
void liberarArbol(Node * root) { if (root == NULL) { return; } liberarArbol(root->left); liberarArbol(root->right); free(root); }
```

Esta función recorre el árbol en postorden y libera la memoria de cada nodo. La función arbolBinario se puede modificar para llamar a esta función al final del programa:

```
scss
Copy code
int arbolBinario(){ // Insertar 50000 nodos aleatorios en el árbol for (int i = 0; i < 50000; i++) { int value = rand() % 50000; root = insertNode(root, value); } // Recorrer el árbol en orden e imprimir los valores de los nodos printf("\n\n\nRecorrido en orden del árbol:\n\n"); inOrderTraversal(root); printf("\n\n\nRecorrido en preorden del árbol:\n\n"); preorden(root); printf("\n\n\nRecorrido en post del árbol:\n\n"); postorden(root); printf("\n"); // Liberar la memoria asignada dinámicamente liberarArbol(root); return 0; }
```

Con esta modificación, se deberían eliminar todos los memory leaks en el programa.

En el código que proporcionaste, la matriz x2 no está inicializada, lo que significa que está utilizando valores no inicializados al imprimir su contenido.

Para solucionar este problema, necesitas inicializar la matriz x2 antes de imprimir su contenido. Puedes hacerlo asignando un valor a cada elemento de la matriz en un bucle for anidado, como se muestra a continuación:

scss

Copy code

```
for (f = 0; f < M; f++) { for (cc = 0; cc < N; cc++) { x2[f][cc] = (f + 1) * (cc + 1); } }
```

Esto asignará un valor a cada elemento de la matriz x2 y eliminará el error de "uninitialized value" que estás viendo en la salida de Valgrind.