```c
#include <inttypes.h>
#include <stdio.h>
#include <stdlib.h>

#define REPEAT(n)   for (uintmax_t _dcntr = (uintmax_t)n; _dcntr--;)

typedef struct ItemBody {
    char        description[24];
    uint32_t    price_in_usd;
    uint32_t    days_until_expiration;
} Item;

static Item const example_items[] = {
    {   "Premium Milk", 12,  1 },
    {   "Toast Breads",  6,  2 },
    {    "Feta Cheese",  8, 24 },
    {  "Standard Milk",  3,  8 },
    { "Salty Crackers",  4, 52 },
    {   "Orange Juice",  7, 30 },
    {  "Chocolate Bar",  2, 44 }
};
static uint32_t const n_example_items =
    sizeof(example_items) / sizeof(example_items[0]);

typedef struct BSTNodeBody {
    Item const*         item;
    struct BSTNodeBody* left;
    struct BSTNodeBody* right;
} BSTNode;

static BSTNode* insertItem(
    BSTNode* node, Item const* const item
);
static void printItemsAbovePriceAscending(
    BSTNode const* node, uint32_t const price_threshold
);

int main(void) {
    Item const* item    = &example_items[0];
    BSTNode* root       = insertItem(root, item++);

    REPEAT(n_example_items - 1)
        insertItem(root, item++);

    printItemsAbovePriceAscending(root, 5);

    return 0;
}
```

The above C code will create a Binary Search Tree of a list of items, and print the items above a certain price threshold in ascending order, where there are n items in total and $k < n$ items above the threshold.

1. Implement a recursive `insertItem` function.
2. Implement the `printItemsAbovePriceAscending` function.
   HINT: Implement recursive `search` and `traverse` functions first. The search function must be able to return the item with the lowest price that is greater than or equal to the threshold.
3. Implement a third function that uses **the same BST** to print all the items with an expiration date due to less than d, in any order.