

CSI 333 – Programming at the Hardware-Software Interface

Two Functions for Linked Lists

Handout 5.2

Program Example 8:

Explanation: In the following C functions, we assume that each node is of type `struct node` defined as follows:

```
/* Struct for each node in the linked list. */
struct node {
    int value;
    struct node *next;
};
```

Two functions are shown in this handout. The actions carried out by these functions are as follows.

1. Given a pointer (`h`) to the first node of a list, the first function (`print_list`) traverses the list and prints the integer values stored in the nodes in the order of their appearance in the list.
2. Given a pointer (`h`) to the first node of the list and an integer value (`x`), the second function (`search_list`) traverses the list and returns a pointer to the first node which contains the value specified by `x`. If no node in the list contains the value specified by `x`, the function returns `NULL`.

```
void print_list(struct node *h) {

    /* Prints the values stored in the nodes of the list */
    /* pointed to by h.                                     */

    if (h == NULL) {
        printf("The list is empty.\n");
    }
    else {
        printf("Values in the list are:\n");
        while (h != NULL) {
            printf("%d\n", h->value);
            h = h->next;
        }
    }
} /* End of print_list */
```

(over)

```
struct node *search_list(struct node *h, int x) {

    /* Returns a pointer to the first node which contains the value */
    /* given by x. If there is no such node, the function returns */
    /* NULL. */

    while (h != NULL) {
        if (h->value == x)
            return h;
        h = h->next;
    }
    /* Here, there is no node with value x. */
    return NULL;
} /* End of search_list */
```