CS23710 Worksheet 6 - (Assignment Three for CS23710)

David Price

March 12, 2001

1 Introduction

This worksheet is intended to provide practice with the use of dynamically created structures.

ALSO NOTE: There are diagrams to accompany this worksheet which are ONLY available as photocopy sheets.

2 The Problem — A Retail Shop

You are required to write a program to simulate the operation of a small retail greengrocery shop. Otherwise known as **"Carol's Cabbage Rides Again, The C Version"**.

The shop has only one cash till and all the products it sells can be classified as either **vegetables** or **fruit**.

2.1 The Stock in Carol's Shop

The shop keeps a record of all the products it has in stock in a tree data structure where each node is a record of exactly the same format. The nodes are to be held as C structures. Each structure has precisely seven fields.

The first field is used to hold a name, the second to hold the units in which the item is sold; both the two fields are character arrays.

The third, fourth and fifth fields, all integers, are used to hold the current stock, the level at which new stock is ordered and the unit price respectively.

The final two fields are both pointers to other stock item structures. The first is used to point at specific varieties of a product. For instance in the example on the diagram attached, the potatoes node has two specific varieties of potato hanging off it, namely, "maris piper" and "lincolnshire whites". This "variety" pointer is set to NULL at the actual leaf nodes which indicate something a customer can actually buy. The second pointer is used to point to other, unrelated, products or product groupings. This second pointer is set to NULL when there are no other products.

The structure MUST be defined in C to match that at the top of the diagram.

2.2 The Cash Till

Carol's shop has only one cash till. This must be defined as a structure with twelve integer fields. Four of the fields are used to hold the number of fifty, twenty, ten and five pound notes respectively. The other eight fields are used to hold the count of coins in each of two pound, one pound, fifty pence, twenty pence, ten pence, five pence, two pence and one pence categories.

3 The Problem — Part One

You should construct a program which uses **malloc** to create the tree of structures and populates them with the stock held in the shop. You should then write a separate function which when provided with a pointer to the top to the stock tree, traverses the tree "depth first" and prints out a summary of all the stock.

4 The Problem — Part Two

Create a function, based on that from part one, which traverses the tree and asks a customer how many products of each type they would like to buy. The function should modify the stock count for each item as the customer buys products and should not sell more products than are in stock. The function should provide as its return value the number of pence spent by the customer.

5 The Problem — Part Three

You should now create the structure for the till and fill it with money. The program should request funds from the customer who can only pay in fifty pound notes ! Based on the amount the customer has spent, the program should calculate the amount of change in terms of cash actually in the till. Attempt to provide the change in the highest denomination notes or coins possible.

6 Effort Allocated to the Worksheet and Assessment for CS23710 Students and Submission Date

This worksheet will be assessed as the third contribution to your mark for the CS23710 module.

You are expected to use the time of the TWO demonstrated practicals for which you are registered during the weeks starting 12th March and 19th March (to get you going), plus up to around ten hours of your own time. Remember that you can consult the Computer Science Advisory service if you have problems.

During the week starting 26th you will be given another worksheet by Helen Fuell.

All students will not necessarily complete all the parts of this worksheet in the time available, perfect answers to the first two parts accompanied by the write-up would still theoretically allow a student to gain a mark of 2(1) quality.

You are required to hand in a listing, including comments, of this worksheet solution together with a short two page description of how the program works. You are also expected to hand printout of a run of the program captured perhaps using “script”. The printout can be on lineprinter paper rather than printed on a laser printer if you wish, thus saving you money.

We expect you to use ANSI ’C’ features and we will penalize breaches of this requirement. You should make sure that your submission is adequately fixed together and clearly bears your name. You should also affix to the front a completed copy of the departments ”Declaration of Originality” which is available from our web site.

You are required to hand in your submission on Monday 26th March between 14:00 and 16:00 into the hand-in facility in the advisory office.

The demonstrators will keep a record of who attends the practical.

7 A Useful Simple Program

```
/* This is an example C Program using malloc and pointers */

/* Version Two */

#include <stdlib.h>
#include <errno.h>

struct course_item { char name[20];
                     int  credit;
                     struct course_item * item_part;
                     struct course_item * alternate;
                   };
print_tree(struct course_item * ci_ptr)
{
    printf("item %s has %d credits\n",(*ci_ptr).name, (*ci_ptr).credit);
    if ( (*ci_ptr).item_part != NULL ) print_tree( (*ci_ptr).item_part);
    else printf("which has no sub-items\n\n");
    if ( (*ci_ptr).alternate != NULL ) print_tree( (*ci_ptr).alternate);
}

struct course_item * make_node(char text[], int amount,
struct course_item * left_ptr, struct course_item * right_ptr)
{ struct course_item * temp_ptr;

temp_ptr = malloc( sizeof (struct course_item) );
strcpy( temp_ptr->name , text );
        temp_ptr->credit = amount;
        temp_ptr->item_part = left_ptr;
        temp_ptr->alternate = right_ptr;
```

```
return temp_ptr;
}


main ()
{
 struct course_item * new_degrees_pointer;


 new_degrees_pointer = make_node("new degrees",0,
     make_node("painting",0,NULL,
          make_node("computing",10,
              make_node("design",10,NULL, make_node("programming",10,NULL,NULL))
                                     ,NULL)),NULL);

 print_tree(new_degrees_pointer);

}
```