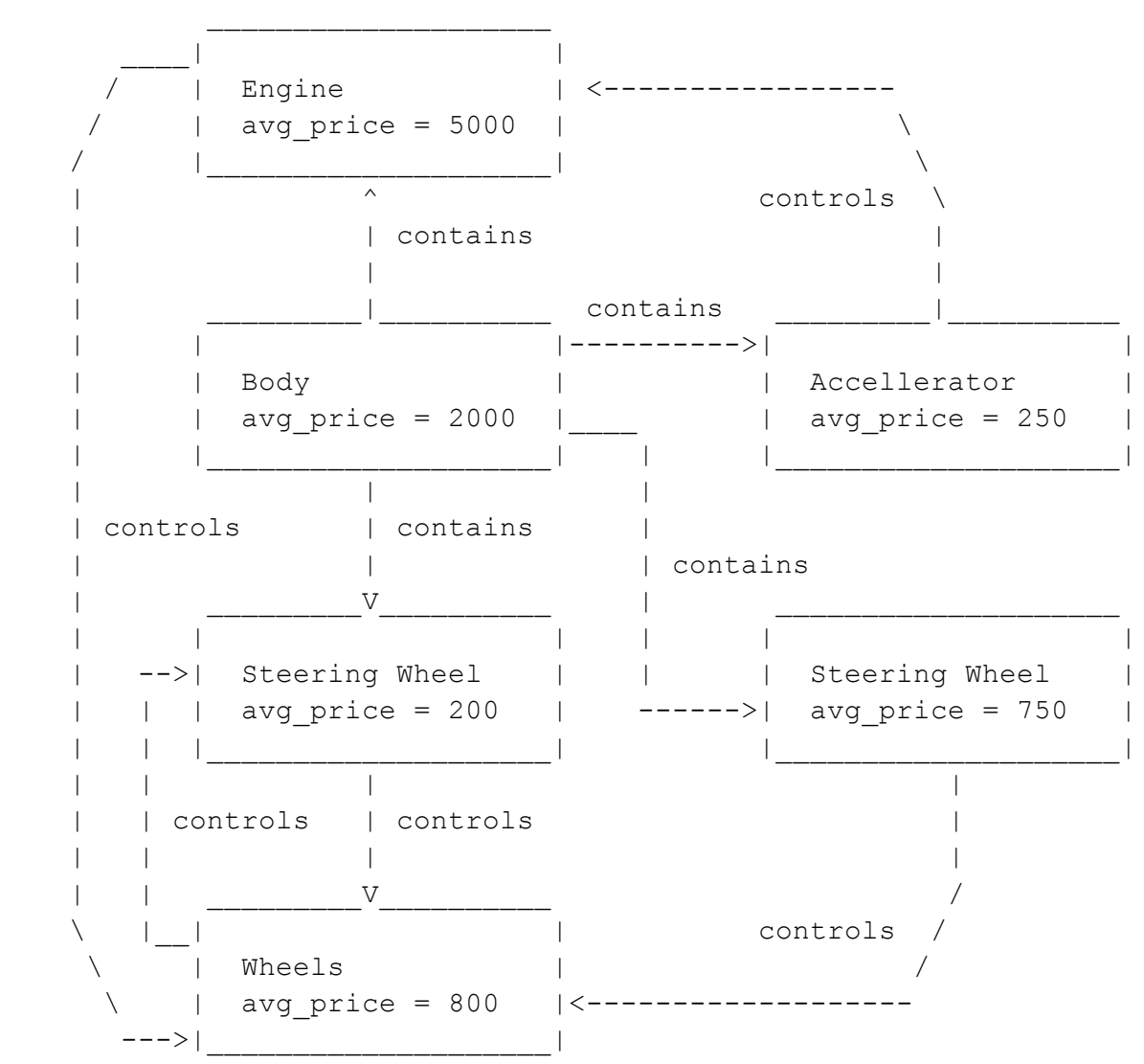


```c
/*
 * C program by Dave Russillo. Made for CS1311.
 * Representation of car parts and their relationships using a network/graph.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>


struct Part {
  char name[80];
  int avg_price;  // in USD
  struct Part **contains;  // pointer to array of pointers
  struct Part *controls;
};


enum Mode {
  NORMAL,
  DEBUG
} mode;
```

```
void populate_network(struct Part *start) {
  void populate_network_helper(struct Part *node,
                               char *name,
                               int avg_price,
                               int contains_amount,
                               struct Part *controls) {
    // assign name
    strcpy(node->name, name);

    // assign average price
    node->avg_price = avg_price;

    // initialize contains array
    struct Part **contains = malloc(sizeof(struct Part*) * contains_amount);
    for(int i = 0; i < contains_amount; i++) {
      // initialize each element
      contains[i] = malloc(sizeof(struct Part));
    }
    // assign contains
    node->contains = contains;

    // assign controls
    node->controls = controls;

    if(mode == DEBUG) {
      printf("New part added---name: %s---location: %p\n", node->name, node);
    }
  }
  // Body: Contains Engine, Accellerator, Braking System, and Steering Wheel. Controls
nothing.
  populate_network_helper(start, "Body", 2000, 4, NULL);
  // Engine: Contains nothing. Controls Wheels.
  populate_network_helper(start->contains[0], "Engine", 5000, 0, (struct
Part*)malloc(sizeof(struct Part)));
  // Wheels: Contains nothing. Controls Body.
  populate_network_helper(start->contains[0]->controls, "Wheels", 800, 0, start);
  // Accellerator: Contains nothing. Controls Engine.
  populate_network_helper(start->contains[1], "Accellerator", 250, 0, start->contains[0]);
  // Braking System: Contains nothing. Controls Wheels.
  populate_network_helper(start->contains[2], "Braking System", 750, 0,
start->contains[0]->controls);
  // Steering Wheel: Contains nothing. Controls Wheels.
  populate_network_helper(start->contains[3], "Steering Wheel", 200, 0,
start->contains[0]->controls);

  // check if all nodes exist
  if(mode == DEBUG) {
    if(start != NULL &&  // Body
       start->contains[0] != NULL &&  // Engine
       start->contains[1] != NULL &&  // Accellerator
       start->contains[2] != NULL &&  // Braking System
       start->contains[3] != NULL &&  // Steering Wheel
       start->contains[3]->controls != NULL) {  // Wheels
      printf("All Nodes successfully populated\n");
```

```
    } else{
      printf("ERROR: One or more Nodes were not successfully initialized\n");
      exit(1);
    }
    printf("\n");
  }
}


void print_network(struct Part *start) {
  void print_network_helper(struct Part *node) {
    // name and price
    printf("The part named '%s' has an average price of $%d", node->name,
node->avg_price);
    if(mode == DEBUG) {
      printf(" and it is located at %p", node);
    }
    printf(".\n");

    // contains
    if(node->contains[0] != NULL) {
      printf("    It contains the following parts:\n");
      for(int i = 0; node->contains[i] != NULL; i++) {
        printf("        - %s", node->contains[i]->name);
        if(mode == DEBUG) {
          printf(" (at %p)", node->contains[i]);
        }
        printf("\n");
      }
    }

    // controls
    if(node->controls != NULL) {
      printf("    It controls the part '%s'", node->controls->name);
      if(mode == DEBUG) {
        printf(" (located at %p)", node->controls);
      }
      printf(".\n");
    }
    printf("\n");
  }
  // call on each node
  print_network_helper(start);  // Body
  print_network_helper(start->contains[0]);  // Engine
  print_network_helper(start->contains[1]);  // Accellerator
  print_network_helper(start->contains[2]);  // Braking System
  print_network_helper(start->contains[3]);  // Steering Wheel
  print_network_helper(start->contains[3]->controls);  // Wheels
}


void print_network_ascii(void) {
  printf(
        "\n----------ASCII Representation of network----------\n"
        "                 ____________________\n"
```

```
        "        ____|                    |\n"
        "      /     |  Engine            | <-----------------\n"
        "     /      |  avg_price = 5000  |                    \\\n"
        "    /       |____________________|                     \\\n"
        "    |                ^                        controls  \\\n"
        "    |                | contains                          |\n"
        "    |                |                                   |\n"
        "    |        _________|__________  contains   _________|__________\n"
        "    |       |                    |---------->|                    |\n"
        "    |       |  Body              |           |  Accellerator      |\n"
        "    |       |  avg_price = 2000  |____       |  avg_price = 250   |\n"
        "    |       |____________________|    |      |____________________|\n"
        "    |                |                |\n"
        "    | controls       | contains       |\n"
        "    |                |                | contains\n"
        "    |        _________V__________      |       ____________________\n"
        "    |       |                    |     |      |                    |\n"
        "    |   -->|  Steering Wheel     |     |      |  Steering Wheel    |\n"
        "    |   |  |  avg_price = 200    |     ------>|  avg_price = 750   |\n"
        "    |   |  |____________________|            |____________________|\n"
        "    |   |            |                                  |\n"
        "    |   | controls   | controls                         |\n"
        "    |   |            |                                  |\n"
        "    |   |   _________V__________                       /\n"
        "    \\   |__|                    |           controls  /\n"
        "     \\      |  Wheels            |                    /\n"
        "      \\     |  avg_price = 800   |<-------------------\n"
        "       --->|____________________|\n\n");
}


int main(int argc, char *argv[]) {
  if(argc == 2 && strcmp(argv[1], "debug") == 0) {
    mode = DEBUG;
    puts("-----DEBUG MODE-----");
  } else if(argc == 1) {
    mode = NORMAL;
  } else {
    puts("Invalid arguments.");
  }

  struct Part *start = malloc(sizeof(struct Part)); // declare and initialize start
  populate_network(start);
  print_network(start);
  print_network_ascii();

  return 0;
}
```