C Review 4
Answers

Question 1

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

typedef struct record record;
struct record {
   char name[32];
   int score;
};

typedef struct node node;
struct node {
   record data;
   node *next;
};

void print(node *lst);
node *find(node *lst, const char *name);
void destroy (node *lst);
int insert (node **plst, const char *name, int score);
size_t delete_all (node **plst, const char *name);


/** 1.a) Printing list */
void print(node *lst){
   node *p;
   for (p = lst; p != 0; p = p->next){
       printf("%s %d \n", p->data.name, p->data.score);
   }
}

/** 1.b) Standard Traversal*/
node *find(node *lst, const char *name){
   node *p;
   for (p = lst; p != 0; p = p->next){
       if (strcmp(p->data.name, name) == 0)
           return p;
   }
   return 0;
}

/** 1.c) Destroy list  */
void destroy(node *lst){
   node *p, *q;
   for (p = lst; p != 0; p = q){
       q = p->next; /* need to remember previous pointer */
       free(p);
   }
```

```
}

/** 1.d) Insert record into list */
int insert (node **plst, const char *name, int score){
   node *newNode = malloc(sizeof(node));
   if (newNode == 0){
       return 0;
   }

   strcpy(newNode->data.name, name);
   newNode->data.score = score;

   newNode->next = *plst;
   *plst = newNode;
   return 1;
}

/** 1.e) Delete all in the list
The diagram for the thought process: https://goo.gl/photos/q2fsJjkomAA7awVr6 */
size_t delete_all (node **plst, const char *name){
   node **tracer;
   size_t counter = 0;

   for (tracer = plst; *tracer != 0;){
       if (strcmp((*tracer)->data.name, name) == 0){
           node *tmp = *tracer;
           *tracer = tmp->next;
           free(tmp);
           counter++;
       } else {
           tracer = &(*tracer)->next;
       }
   }
   return counter;
}

int main(void){
   node *head = 0;
   int a[] = {3, 2, 7, 2, 3, 3, 1};
   const char *name[] = {"homer", "bart", "lisa", "ned", "marge", "waylon", "monty"};
   size_t i;
   for (i = 0; i < sizeof(a) / sizeof(a[0]); i++){
       if (!insert(&head, name[i], a[i])){
           return 1;
       }

   }
   print(head);
   delete_all(&head, "monty");
   print(head);
   return 0;
}
```

Question 2

```c
typedef struct record record;
struct record {
   char name[32];
   int score;
};

/** 2.a) Descending order of scores */
int cmp(const void *p, const void *q){
   const record *pp = p;
   const record *qq = q;
   return qq->score - pp->score;
}

/** 2.b) ascending order of names */
int cmp (const void *p, const void *q){
   const record *pp = p;
   const record *qq = q;
   int n = strcmp(pp->name, qq->name);
   if (n != 0){
       return n;
   }
   return qq->score - pp->score;
}
```

Question 3

---

a[n] same thing as *(a + n)

```c
/** 3.a) Indicate if it's valid, and what is the output */
char a[] = "hello";
char *p = "world";
p = a + 2; /* @ */
printf("%s", p);
```

valid; llo

```c
/** 3.b) Indicate if it's valid, and what is the output */
char a[] = "hello";
*a = *(a + 1); /* @ */
printf("%s", a);

a[0] = a[1];
```

valid; eello

```c
/** 3.c) Indicate if it's valid, and what is the output */
char a[] = "hello";
char *p = "world";
*(p + 1) = a[0]; /* @ */
printf("%s", p);
```

```
p[1] = a[0];
invalid, can't change constant
```

Question 4

```
/** 4.a) Describe what x is */
int (*x)[2];
x is a pointer to array of 2 ints

/** 4.b) Describe what x is */
int (*x)(float);
x is a pointer to a function that takes a float
and returns an int

/** 4.c) Describe what x is */
int (*x[2])(float);
x is an array of 2 pointers to functions that take
a float and return an int
```