Homework #2  
Due: Monday, Feb. 23 at 2pm

1. (5 pts) Write a definition for a structure that represents a student. A student should have name, gpa value and a list of course taken in a given semester.

2. (6 pts) Write a binary search function that searches for a word in a sorted array of words.

3. (5 pts) Assume we are executing the following program on a 32-bit machine:

   (a) What is the output?
   (b) If we change p’s declaration to be: double *p; and use the type casting with malloc to be (double *), what is the output?
   (c) In both cases, how many bytes are freed by calling “free(...)”?

   ```c
   int main() {
       int *p;
       p = (int *) malloc(32);
       printf("%d\n", sizeof(p));
       free(p);
       return 0;
   }
   ```

4. (4 pts) What is printed by the program shown below? What is the name of the sequence of numbers that are printed?

   ```c
   #include <stdio.h>
   #include <stdlib.h>
   typedef struct node* pointer;
   struct node {
       int data;
       pointer next;
   };
   int main(void) {
       pointer x, y, t;
       x = malloc(sizeof(struct node));
       y = malloc(sizeof(struct node));
       x->next = y;
       x->data = 1;
       y->next = x;
       y->data = 1;
       t = x;
       int counter = 0;
       for( counter = 0; counter < 10; counter++) {
           t->data = x->data + y->data;
           printf("%d\n", t->data);
           t = t->next;
       }
       return 0;
   }
   ```
5. (6 pts) A node for a binary search tree is defined as follows:

```c
struct bst_node {
    char * data;
    struct bst_node * right;
    struct bst_node * left;
}
```

(a) Write a function that given
- a pointer to the root (possibly equal to NULL for an empty tree) and
- a string
adds a new node in the appropriate position of the BST.
Assume that the left child stores a value that is <= to the root and the right child stores a value that is > than the root.

(b) Write a function that removes the node storing the smallest value.

6. (4 pts) Consider the following C program:

```c
#include <stdlib.h>

int main(int argc, char** argv) {
    float x[3][4];
    float *a = &(x[3][0]);
    float *b = a;
    b--;
    float *c = &(x[2][4]);
    float *d = &(c[1]);
    int e = x[3] - x[1];
    int f = &(x[3]) - &(x[1]);
    int g = &(x[3][0]) - &(x[1][0]);
    float *h = x[3] - 2;
    return 0;
}
```

Assume that this program runs on an architecture where pointers are 4 bytes large, and the sizes of char and float are 1 and 4 bytes, respectively. Further assume that the value of pointer variable x is the address 100. What are the values of variables 'a' through 'h' just before the return statement?

Warning 1: this program will generate some warning if you compile it - these are not errors and for the purpose of this exercise should be ignored.

Warning 2: this is a hard problem that tests your understanding of pointers. Explain your answers to help us understand how you arrived at them.

How to submit: Your submission should be on paper and handed in at the beginning of the class on Wednesday, Feb. 19. It has to be typed (no hand-written assignments will be accepted). You should answer the questions in order. Read the questions carefully and make sure you are answering the question that is asked. You should write and run most (if not all) of the code that you write to make sure that it is correct.