CSE2031

Lab 5
Summer 2016
Practice Questions

Here are some practice questions to test your C abilities

1. Modify the `qsort.c` program below so that low, high, and middle are pointers to array elements rather than integers. The `split` function will need to return a pointer, not an integer.

```
/* Sorts an array of integers using Quicksort algorithm */

#include <stdio.h>
#define N 10

void quicksort(int a[], int low, int high);
int split(int a[], int low, int high);

int main(void)
{
  int a[N], i;

  printf("Enter %d numbers to be sorted: ", N);
  for (i = 0; i < N; i++)
    scanf("%d", &a[i]);
  quicksort(a, 0, N - 1);

  printf("In sorted order: ");
  for (i = 0; i < N; i++)
    printf("%d ", a[i]);
  printf("\n");

  return 0;
}
void quicksort(int a[], int low, int high)
{
  int middle;

  if (low >= high) return;
  middle = split(a, low, high);
  quicksort(a, low, middle - 1);
  quicksort(a, middle + 1, high);
}

int split(int a[], int low, int high)
{
  int part_element = a[low];

  for (;;) {
```

```
    while (low < high && part_element <= a[high])
      high--;
    if (low >= high) break;
    a[low++] = a[high];

    while (low < high && a[low] <= part_element)
      low++;
    if (low >= high) break;
    a[high--] = a[low];
  }
  a[high] = part_element;
  return high;
}
```

2. Modify the maxmin.c program below so that the `max_min` function uses a pointer instead of an integer to keep track of the current position in the array.

```
/* Finds the largest and smallest elements in an array */
#include <stdio.h>
#define N 10
void max_min(int a[], int n, int *max, int *min);

int main(void)
{
  int b[N], i, big, small;

  printf("Enter %d numbers: ", N);
  for (i = 0; i < N; i++)
    scanf("%d", &b[i]);
  max_min(b, N, &big, &small);
  printf("Largest: %d\n", big);
  printf("Smallest: %d\n", small);
  return 0;
}
void max_min(int a[], int n, int *max, int *min)
{
  int i;

  *max = *min = a[0];
  for (i = 1; i < n; i++) {
    if (a[i] > *max)
      *max = a[i];
    else if (a[i] < *min)
      *min = a[i];
  }
}
```

3. Write the following function:

   ```
   Bool search (const int a[], int n, int key);
   ```

   `a` is an array to be searched, *n* is the number of elements in the array, and key is the search key. `search` returns true if key matches some element of `a` and false if it doesn't. Use pointer arithmetic—not subscripting—to visit array elements.

4. Write the following function:

   ```
   void find_two_largest (int a[], int n, int *largest, int *second_largest);
   ```

   When passed an array `a` of length *n*, the function will search array `a` for its largest and second_largest elements.

5.
   a) Write a program that reads a message, then checks whether it's a palindrome (the letters in the message are the same from left to right as from right to left):

      Enter a message: `He lived as a devil, eh?`
      Palindrome

      Enter a message: `Madam, I am Adam.`
      Not a palindrome.

      Ignore all characters that aren't letters. Use integer variables to keep track of positions in the array.

   b) Revise the program to use pointers instead of integers to keep track of positions in the array.