Maximum Marks: 30 16th Mar, 2014 Time Allowed: 90 minutes

Question 1: [12 Marks]

In each of the following, indicate whether the operations result in **inaccessible memory**, a **dangling pointer**, **both**, or **neither**. Justify your answer(s).

```
char * n = "Ali";
float * pPi = new float;
                                         char * m = new char[20];
float * pPi2 = new float;
*pPi = 3.1415;
                                         strcpy(m,n);
                                         char * temp = n; n = m; m= temp;
pPi2 = pPi;
                                         delete [] m;
                                         m = nullptr;
int * n = new int[10];
                                         void alloc(int *a, int size) {
*n = 0;
                                               a = new int [size];
for (int i = 0; i<10; i++) {
      *(n+i) = *(n);
                                         int main(){
}
                                               int * arr;
delete n;
                                               alloc(arr, 10);
n = nullptr;
                                               arr[0] = 10;
                                               delete []arr;
                                               arr = nullptr;
                                         }
int * cube (int * a) {
                                         int meaning = 42;
                                         int *life = &meaning;
      int s = *a**a**a;
                                         int **universe = &life;
      return &s;
                                         int ***everything = &universe;
}
                                         cout << ***everything <<endl;</pre>
int main(){
                                         delete life;
      int i = 10;
                                         life = nullptr;
      int j = *cube (&i);
                                         universe = nullptr;
                                         everything = nullptr;
      cout << j <<endl;</pre>
}
```

Maximum Marks: 30 16th Mar, 2014 Time Allowed: 90 minutes

Question 2: [8 Marks]

Write a function called **allPositionsOfSub** which receives two cstrings (character arrays) called **str** and **sub** respectively. The goal of the function is to return all the positions in **str** where there is an occurrence of the string **sub**. For example, if **str** is the string "**Posing a possible post of possibilities**", and sub is the string "**pos**", your function should return an array containing the numbers [**0**, **9**, **18**, **26**, **-1**], as these are the indices in the string **str** where an instance of the string "**pos**" begins and **-1** is the end marker.

Please note the following:

- The array that is being returned should be of size exactly one more than the number of times sub occurs in **str** (for example, it is exactly of size **5** (one more than the number of occurrences in the case shown above). In case sub does not occur in **str**, your function should return **NULL**.
- Make sure there are no memory leaks in your function.
- You don't need to take any inputs or do any outputs.
- Pay special attention to the function arguments and their types.
- You do not need to write the main function. Only the code for allPositionsOfSub is required.

Question 3: [10 Marks]

Write a function called transpose, which converts a dynamic two dimensional array into its transpose. The array will always be rectangular, and dimensions m x n shall also be passed to the function as parameters. After the execution the function must have deleted the older array and created a new one in its place, as well as copied the data appropriately. The following example shows a two dimensional array of dimension 3 x 4, and its corresponding 4 x 3 transpose.

