BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI (RAJ.)

Second Semester 2017-18 CS F111 Computer Programming

LABORATORY SESSION #10

(Dynamic Memory Allocation; Array of Pointers)

1. Take a look at the ATM branches problem you worked on during last week's lab (question #3 of Lab Sheet #9). Copy the program you wrote for this question into your current working directory and rename it as `10.2.c`. Now consider the following modification of the problem statement:

> It was observed that there were few ATM branches that had very few transactions, but others that had several. The file available at `/home/share/10.2.txt` has the data – the first row has the number of branches, while the first number in each of the other rows (representing a branch) indicate the number of transactions recorded for that branch, followed by the actual transaction values.

Now modify the code in `10.2.c` in a way the new data can be captured – using a dynamically allocated array rather than an m x n matrix allocation you had done earlier. Use an array of pointers and dynamically allocate memory to store the data for `m` branches.

2. In the class, we discussed selection sort technique that sorts an array of elements. Insertion sort is another technique for sorting whose algorithm is given below:

```
1. Read N, Arr
2. FOR i = 1 to N-1
       Key = Arr[i]
       j = i - 1
       WHILE ( j >= 0 and Arr[j] > Key)
            Arr[j+1] = Arr[j]
            j = j - 1
       Arr[j+1] = Key
3. Output Arr
4. Stop
```

a. Understand the algorithm by taking a sample array of elements {8,5,3,9,4}. Write down in your lab notebook the stepwise results of applying the above algorithm on this input.

b. Translate this algorithm into a C function which has the following prototype:

```
void insertionSort(int arr[], int num);
```

Test the function by calling it in main() with sample data.

3. Two words/phrases are said to be anagrams if one of them can be rearranged to give the other. Write a function **`isAnagram()`** that takes two strings as arguments and tells whether the strings are anagrams of one another. For example, the function should return a value of 1 for pair of string inputs such as “great” and “grate”; “Listen” and “siLent”; “rail safety” and “fairy tales”. It returns a 0 for string inputs such as: “night” and “knight”; “great” and “Grate”; “Madam Curie” and “Radium came”.

This problem can be solved in multiple ways. First spend some time thinking of the algorithm, write the pseudo code and then go ahead with the implementation.

4. Identify the errors in each of the following code snippets and fix them:

a.
```
char *pt;
scanf("%s", pt);
printf("%s", pt);
```

b.
```
int x[20] = { 1,2,3,4};
int *p = x + 15;
for(int i=0; i >= -20 ; i--)
     printf("%d", p[i]);
```

c.
```
char *months[] = {"January", "February", "March"};
*(months)[1] = 'A';
1[months][2] = 'E';   // There is no typo here!
```

Additional Practice Questions:

Modify the function you wrote for question #3 above so that the new function called **`isAnagram2()`** ignores the case of the letters of the alphabet of the strings that are compared. For example, this function would return a value 1 even for the two strings “Madam Curie” and “Radium came”.

Next, write the third version of this function **`isAnagram3()`** that is not only case-insensitive, but also disregards any intervening spaces. For instance, this function will recognize each of the pairs such as “dormitory” and “dirty room”, “Slot Machines” and “Cash lost in me” as anagrams.