CMPT 125 Week 4 Lab Work

C++ Vectors

1. Consider the following C++ program. Define the **`swapVectors`** function that swaps its vector arguments so that the program works without any errors. Sample run output is shown to help you understand the problem.

Remark: You should not have any loop in your **`swapVectors`** function. You should be able to swap without using any loop. Explain why?

```
void printVector(const vector<int> &v)
{
   for (int i = 0; i < v.size(); i++)
      cout << v[i] << "    ";
   cout << endl;
}
int main()
{
   srand(time(0));
   vector<int> v1, v2;
   int n1 = rand() % 5 + 1, n2 = rand() % 5 + 1;
   for (int i = 0; i < n1; i++)
      v1.push_back(rand()%31 - 15);
   for (int i = 0; i < n2; i++)
      v2.push_back(rand()%31 - 15);
   cout << "Initially v1 = ";
   printVector(v1);
   cout << "and v2 = ";
   printVector(v2);
   swapVectors(v1, v2);
   cout << "After swapping, v1 = ";
   printVector(v1);
   cout << "and v2 = ";
   printVector(v2);
   system("Pause");
   return 0;
}
```

Sample Run Output

```
Initially v1 = [10, -8, -8, 12, 14]
and v2 = [-12, 8, 12, 10, -7, 10, -11, 8, 5]
After swapping, v1 = [-12, 8, 12, 10, -7, 10, -11, 8, 5]
and v2 = [10, -8, -8, 12, 14]
Press any key to continue . . .
```

2. Define a function that takes a vector of integers and removes the even number elements of the vector.

Hint:- You may answer this question in one of two ways: using the erase method or using the assignment operator.

3. Write a C++ program that constructs an empty vector of int data type, pushes back five random integers in the range [0, 5] in to the vector, prints the elements of the vector, duplicates each element of the vector as many times as the value of the element itself, and finally prints the modified vector.

For example, if the vector is initially [2, 1, 4, 0, 5] then at the end the element 2 should be duplicated twice, 1 should be duplicated once, 4 should be duplicated 4 times, 0 should be duplicated 0 times and 5 should be duplicated 5 times; which means the vector should be modified to [2, 2, 1, 4, 4, 4, 4, 5, 5, 5, 5, 5] at the end.

4. Define a function named **`distinctElements`** that takes a vector of int data type and returns a new vector containing the distinct elements of the argument. Your function should not modify the argument. For example if the vector argument contains the elements [2, 4, 2, 6, 3, 6] then your function must return a new vector containing [2, 4, 6, 3].

5. Define a function named **`isDistinct`** that takes a vector of integers and returns true if the vector contains distinct elements; otherwise returns false. Your should not have any loop in your function block; instead you must use the **`distinctElements`** function defined in question #4 above.

6. During the lecture, we have demonstrated the quick sort algorithm using a C++ static array variable. Re-write the example using a C++ vector instead of a C++ static array.

7. Consider the following C++ program.

```
#include<iostream>
#include<vector>
usingnamespacestd;

void printVector(const vector<int> &x)
{
   for (int i = 0; i < x.size(); i++)
      cout << x[i] << "  ";
   cout << endl;
}
void insertIncreasing(vector<int> &a, const int x)
{
   //Fill your code here
}
int main()
{
   vector<int> a;
   for (int i = 0; i < 5; i++)
   {
      int num = rand() % 11 - 5;
      cout << "Inserting " << num << endl;
      insertIncreasing(a, num);
      printVector(a);
   }
   system("Pause");
   return 0;
}
```

Sample Run Output

```
Inserting 3
3
Inserting 8
3   8
Inserting -2
-2   3   8
Inserting 6
-2   3   6   8
Inserting 0
-2   0   3   6   8
```

Define the **`insertIncreasing`** function that takes a vector (whose elements are already arranged in increasing order) and an integer number arguments and inserts the integer argument in the vector in its correct position; so that after the insertion operation, all the elements of the modified vector are arranged in increasing order. You must use a linear algorithm to perform the operation.

8. Define a C++ function named **`insertGrouped`** that takes a vector of integers whose elements are grouped (i.e. its even integer elements are grouped together and its odd integer elements are also grouped together) and an integer value as arguments and inserts the integer argument in the vector such that after the insertion operation, the elements of the modified vector are also grouped. You must use a linear algorithm to perform the task. Use a similar test code as Q7 above.

9. Define a C++ function named **`insertGroupedIncreasing`** that takes a vector of integers whose elements are grouped into even and odd groups and within each group they are sorted in increasing order and an integer value as arguments and inserts the integer argument in the vector such that such that after the insertion operation, the elements of the modified vector are also grouped into even and odd groups and within each group elements are sorted in increasing order. You must use a linear algorithm to perform the task. Use a similar test code as Q7 above.

Recursion

10. Define a C++ recursive function named **reversePrint** that takes a non-negative integer argument and prints the number in reverse. For example: if the argument integer is 1234, then the function must print **4321.**

    Hint:- What should be the base case? When the parameter is equal to zero or when the parameter is less than 10? Choose your base case carefully to make the problem solving easier.

11. Observe that given any integer **x** and a positive integer **y**, the expression $\mathbf{x^y}$ can be written as $\mathbf{x*x^{y-1}}$. Define a recursive function named power that takes an integer **x** and a potive integer **y** and returns **x raised to the power of y**.

12. Define a recursive function named **squares** that takes a positive integer argument **n** and returns the sum, in the order given, of the squares $\mathbf{n^2 + (n-1)^2 + (n-2)^2 + \ldots + 1^2}$. For example, **squares(3)** returns **14** because $\mathbf{3^2 + 2^2 + 1^2}$ is **14**.

13. Define a recursive function named **`isDistinct`** that takes a vector of int data type, start index, and last index and returns true if the vector contains distinct elements; otherwise returns false. Note that the elements of a vector are said to be distinct if each element of the array is different from every other element.

14. Define a recursive function named **`findIndex`** that takes a vector of integers, a start index, and a last index arguments such that the there exists an index **k** in the vector with the condition that all the elements of the vector at indexes 0, 1, 2, …, **k** are even integers and the rest are odd integers. Your function must return the value of the index **k**. The complexity of your algorithm must be logarithmic.

    For example if the vector argument contains the elements [4, 0, 8, 14, 2, 7, 3] then your function must return 4.

15. The mathematical Fibonacci sequence is given by $$\boldsymbol{fib(n)} = \begin{cases} \mathbf{1} & \boldsymbol{if\ n = 0} \\ \mathbf{1} & \boldsymbol{if\ n = 1} \\ \boldsymbol{fib(n-1) + fib(n-2)} & \boldsymbol{if\ n} > 1 \end{cases}$$

    Define a recursive function that takes a non-negative integer argument **n** and returns ***fib(n)***.

16. The critical operation of the algorithm is the addition operation. The complexity of the algorithm for a given value **n** is therefore

    f(n) = number of times the addition operation is performed to calculate ***fib(n)***

    What is the **Big-O** of ***f(n)***?

17. Define a recursive function that takes an array of integers, a start index and a last index and prints the elements of the array in reverse order.

18. Define a recursive function that takes a C++ string, a start index and a last index and prints the string in reverse.

19. Define a recursive function that takes a vector of double data type, a start index, and a last index and returns the maximum element of the array.

20. Define a recursive function that takes a C++ string, a start index and a last index and returns the reverse of the string.

21. Define a recursive function that takes a vector of float data types, a start index, and a last index and checks if the elements of the array are sorted in increasing order. If yes, return true; otherwise return false.

22. Define a recursive function that takes a C++ string, a start index and a last index and returns true if the string is a palindrome and returns false otherwise. Remark: A string is palindrome if it is equal to its reverse.

23. **[Challenge: Beyond the scope of the course]** Define a void function named printPermutations that takes a C++ string argument and prints all the strings formed by permutations of the characters of the argument. For example, if the C++ string argument is **`"abc"`** then your function must print the strings **`"abc", "acb", "bac", "bca", "cab",`** and **`"cba"`**.