Roll No ______________________________ Block No _____________

CS101 End-Semester Examination

General instructions: **Time: 3 hours, Max Marks: 122**

- There are 8 questions in this exam. Write your answers directly on this paper in the spaces provided.
- Do not use a more general type where a simpler type would work (e.g., do not use a **float** type where **int** will work.)
- Follow C++ syntax **strictly** when completing missing parts of code.
- Answers written in pencil will not be corrected.

Useful STL functions:

Vector:

vector<T>::iterator - iterator to traverse a vector of type **T**
v.push_back(element t) : adds element **t** to the back of the vector.
v.size(): returns a **size_t** variable holding the length of v.
v.end() - returns an iterator referring to the element (non-existent yet) that would follow the last element of the vector.

String:

string::npos - Value of **string::npos** is a number that does not correspond to any position in any possible string. It is also greater than any other value returned by **s.find()** function
str.find(s) - Returns the index in **str** where the string or character **s** occurs first
str.substr(i, l) - returns a substring of **str** of length **l** starting at index **i** in **str**
str.substr(i) : returns substring starting from index i till the end. In case of i = length of string, it returns “”.
str.length() - returns the number of characters in the string **str**

Map:

m1[key] = value - inserts an element **value** for the key **key** in the map **m1**.

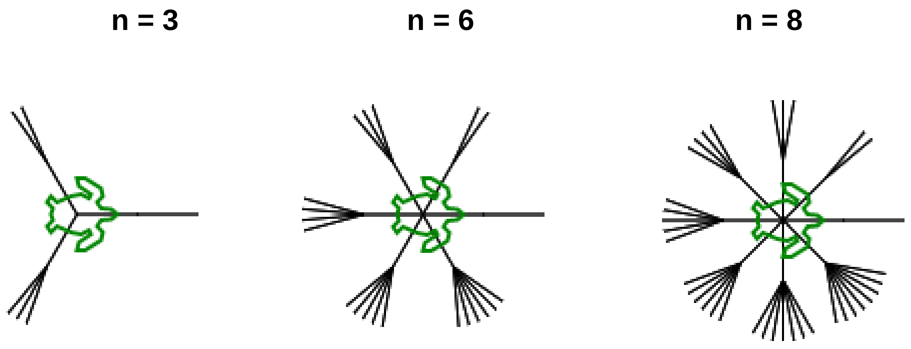
List:

list<T>::iterator - iterator to traverse a list of type **T**.
l1.push_back(element t) - adds element **t** to the back of the list **l1**.
l1.size() - returns a **size_t** variable holding the length of list **l1**.
l1.end() - returns an iterator referring to the element (non-existent yet) that would follow the last element of the list.

Q. No	Marks	Correcting TA	Verifying TA	Q. No	Marks	Correcting TA	Verifying TA
1				5			
2				6			
3				7			
4				8			

Roll No ________________________________ Block No ______________

Q1). [12 Marks]



Qualitative description

i. Each diagram consists of n branches evenly spaced with side length = 30 protruding from a centre.
ii. Further, each ith branch is further split at its end into i sub-branches (call them twigs).
iii. The angle between the branches is independent of the number of twigs.
iv. The length of each twig is fixed and equal to that of the branch i.e., = 30.
v. These twigs are aligned symmetrically about the branch they originate from.
vi. The adjacent twigs have a constant angular separation of 10^o.

a).Fill in the following code that produces the pattern as shown above

```
#include <simplecpp>
main_program{
   turtleSim();
   int i = 1;
   int step = 30;
   int n;
   cin >> n;
   repeat(n){
      forward(step);
      left(_______); //start from the leftmost twig
      repeat(_______){
         forward(step);
         left(180);
         forward(step);
         left(180);
         right(10); // for the angular separation between twigs
      }
      left(________); // align turtle to the branch direction
      left(180);
      forward(step);
      left(180);
       left(________); // set turtle direction right to draw the next branch and
                              its twigs
      i++;
   }
```

Roll No ________________________________ Block No ______________

```
}
```

Answer:
Code:

```
#include <simplecpp>
main_program{
   turtleSim();
   int i = 1;
   int step = 30;
   int n;
   cin >> n;
   repeat(n){
      forward(step);
      left((i-1)*5); //start from the leftmost twig
      repeat(i){
         forward(step);
         left(180);
         forward(step);
         left(180);
         right(10); // for the angular separation between twigs
      }
      left(10+(i-1)*5); // align turtle to the branch direction
      left(180);
      forward(step);
      left(180);
       left(360.0/n); // set turtle direction right to draw the next branch and
                            its twigs
      i++;
   }
}
```

Evaluation instructions: 3 marks per blank
Total: 12 marks

Roll No ______________________________ Block No _____________

Q2). [16 Marks] Given a string s, you have to replace all instances of

i. 'AND' with '&'
ii. 'QN' with '?'

and print the resultant string.

Illustration:
Consider s = **'and_is_AND_but_and_is_not_AND_though_QN_is_?'**
The output must be
'and_is_&_but_and_is_not_&_though_?_is_?'

Following is the code that does the above function. Fill the blanks appropriately.
Note: In 2nd and 5th blank you are required the fill a single character (either '?' or '&').

```
#include <iostream>
#include <string>
using namespace std;
int main(){
  string input;
  cin >> input;
  size_t and_pos, qn_pos;
  while(true){
        and_pos = input.find("AND");
        qn_pos = input.find("QN");
        if(and_pos == string::npos && qn_pos == string::npos){
            cout << input;
            break;
        }
        else{
  // This means that at most one of and_pos and qn_pos equals string::npos

            if(qn_pos < and_pos){
                       /* This can mean that both QN and AND are present and
                           QN is present before AND or that AND is not present
                           at all. */
                cout << input.substr(0, ______) << '_____';
                input = input.substr(______);
            }
            else{
                cout << input.substr(0, _____) << '____';
                input = input.substr(_____);
            }
        }
  }
```

Roll No ________________________________ Block No ______________

}

Answers:

1. qn_pos
2. ?
3. qn_pos + 2
4. and_pos
5. &
6. and_pos + 3

Evaluation instructions:

2 & 5 : 2 marks each

3 marks for each 1,3,4,6

Roll No ______________________________ Block No _____________

Q3). [14 Marks] The program below does the following:

i. Takes input in while loop, and updates a vector.
ii. If input is 0, terminates.
iii. If input is 1, prints the contents of vector in reverse order.
iv. If input is even and not 0, then inserts into vector the sum of input to the last term of
vector. If no term is present in the vector, then just push the input in the vector.
v. If input is odd and not 1, then inserts into vector the difference of last term of vector and
the input.If no term is present in the vector, then just push the input in the vector.

Now fill in the blanks provided:

```
#include <iostream>
#include <vector>
using namespace std;

void print(vector<int> entries){
   size_t len = entries.size();
   for(int i=len-1; i>=0; i--){
        cout << entries[i] << " ";
   }
   cout << endl;
}

int main(){
   vector<int> entries;
   while(1){
        int t;
        cin >> t;
        if(t==0) break;
        else if(t==1) ______________________________;
        else if(t%2==0){
               if____________________________ entries.push_back(t);
               else entries.push_back ______________________________;
        }
        else{
               if ___________________________ entries.push_back(t);
               else entries.push_back ____________________ ;
        }


   }
}
```

Roll No ________________________________ Block No ______________

Answer:

```
#include <iostream>
#include <vector>
using namespace std;

void print(vector<int> entries){
   size_t len = entries.size();
   for(int i=len-1; i>=0; i--){
        cout << entries[i] << " ";
   }
   cout << endl;
}

int main(){
   vector<int> entries;
   while(1){
        int t;
        cin >> t;
        if(t==0) break;
        else if(t==1) print(entries);
        else if(t%2==0){
               if(entries.size()==0) entries.push_back(t);
               else entries.push_back(entries[entries.size()-1] + t);
        }
        else{
               if(entries.size()==0) entries.push_back(t);
               else entries.push_back(entries[entries.size()-1] - t);
        }

   }
}
```

Evaluation instructions:
2+2+4+2+4
Total: 14 marks

Roll No ______________________________ Block No _____________

Q4). [19 Marks] Application Software Cell (ASC) at IIT Bombay manages all the information concerning students' enrolment, grades, SPI, CPI, fees etc. The ASC programmers have heard a lot about the C++ STL knowledge of CS 101 students, and they are reaching out to you for a little help. In this problem, you will implement a few C++ functions that are crucial for ASC's overall working.

ASC maintains a list of all departments in the institute and the list of all students in the department (BTech/MTech/PhD in respective years). See the following declarations to get an idea of how the information is organized.

```
list<string> departments;  // e.g. departments = {"Aero", "Mech", "CS"}

map<string, vector<string> > btech_students;
// e.g. btech_students["EE"] = {"12D070001", …}

map<string, vector<string> > mtech_students;
// e.g. similar to btech_students

map<string, vector<string> > phd_students;
// e.g. similar to btech_students
```

The vector<string> contains roll numbers of students.

(a) Find a student: Implement a function **bool find_student(string roll_number, string dept, string program)**, that returns **true** or **false** depending on whether a student with the specified roll_number/department/program exists in the data structures.

<u>**Template:**</u>

```
bool find_student(string roll_number, string dept, string program) {

        ________________________________ it = departments.begin();
        for(; it != departments.end(); it++) {
                if (*it == dept) {
                        break;
                }
        }
        if (it == __________________________________) {
                // return false if dept is not a valid department
                return false;
```

```
    }

    bool found = false;
    if (program == "btech") {
        for(int i = 0; i < __________________________________; i++) {
            if (_________________________________ == roll_number) {
                return true;
            }
        }
        return false;
    } else if (program == "mtech") {
        for(int i = 0; i < _________________________________; i++) {
            if (________________________________ == roll_number) {
                return true;
            }
        }
        return false;
    } else if (program == "phd") {
        for(int i = 0; i < _________________________________; i++) {
            if (________________________________ == roll_number) {
                return true;
            }
        }
        return false;
    } else {
        return ___________________________________________;
    }
}
```

b). Enrol a student: Implement a function void **enroll_student(string roll_number, string dept, string program)**, which adds a student's roll number into the correct map/list for the correct department/program specified as arguments. If a student already exists with the specified roll_number, department and program, simply return from the function without enrolling the student again. (hint: use the **find_student()** function as defined above).
Assume that the **string program** can take only the values "btech", "mtech", and "phd".

Template:

```
void enroll_student(string roll_number, string dept, string program) {
```

Roll No ________________________________   Block No ______________

```
    if (___________________________________) {
        return;
    }
    if (program == "btech") {
        _________________________________
    } else if (program == "mtech") {
        ________________________________________________
    } else if (program == "phd") {
        __________________________________________________
    }
}
```

Answer
a).

```
bool find_student(string roll_number, string dept, string program) {

        list<string>::iterator it = departments.begin();
        for(; it != departments.end(); it++) {
                if (*it == dept) {
                        break;
                }
        }
        if (it == departments.end()) {
                return false;
                // return false if dept is not a valid department
        }

        bool found = false;

        if (program == "btech") {
                for(size_t i = 0; i < btech_students[dept].size(); i++) {
                        if (btech_students[dept][i] == roll_number) {
                                return true;
                        }
                }
                return false;
        } else if (program == "mtech") {
                for(size_t i = 0; i < mtech_students[dept].size(); i++) {
                        if (mtech_students[dept][i] == roll_number) {
                                return true;
                        }
                }
                return false;
        } else if (program == "phd") {
                for(size_t i = 0; i < phd_students[dept].size(); i++) {
```

Roll No ________________________________ Block No ______________

```
                        if (phd_students[dept][i] == roll_number) {
                                return true;
                        }
                }
                return false;
        } else {
                return false;
        }
}
```

Answer b).

```
void enroll_student(string roll_number, string dept, string program) {
   if (find_student(roll_number, dept, program)) {
        return;
   }
   if (program == "btech") {
        btech_students[dept].push_back(roll_number);
   } else if (program == "mtech") {
        mtech_students[dept].push_back(roll_number);
   } else if (program == "phd") {
        phd_students[dept].push_back(roll_number);
   }
}
```

Marks:

a) 2+2+1+1+1+1+1+1+1+2
b) 4+1+1+1

Roll No ________________________________ Block No ______________

Q5). [14 Marks] Given a number **x** and a sorted array of integers **a**, you need to find whether there exists a pair such that their sum is equal to **x**.

(a). Fill in the blanks in the code to solve this problem. The boolean variable flag should be 0 if such a pair doesn't exist, otherwise it should be 1.

```
#include <iostream>
using namespace std;
int main(){

   int n, x, flag=0;
   cin >> n >> x;
   int a[n];

   for(int i=0;i<n;i++)
        cin >> a[i];

   int l = 0,r = n - 1;

   int num_iter = 0;
   while(l < r){
      num_iter ++ ;
        int cur_sum = a[l] + a[r];

        if(cur_sum == x){
                ________________________________
                break;
        }
        else if(cur_sum < x){
                l = _____________________________;
        }
        else
                r = ______________________________;

   }

   cout << flag << " " << num_iter << endl;
}
```

Roll No ________________________________ Block No ______________

(b). What will be the value of the variable num_iter in the output of this program for the following input?
6 21
-8 -5 4 26 40 53

Answer:
(a). Blank-1: **flag = 1**
Blank-2: **l + 1**
Blank-3: **r – 1**

3 marks each

(b). 4

5 marks

Total - 14 marks

Roll No ______________________________ Block No _____________

Q6). [16 Marks] The inputs to this problem are the positive integers a0, . . . , a9 and b0, . . . , b9. The goal is to calculate
r = (a0 × a1 × · · · × a9)/(b0 × b1 × · · · × b9)

It is known that all ai, bi and r are integers smaller than 2^32. It is possible, however, that the numerator or denominator of the above fraction are larger than 2^32.

a).Now fill in the blanks in the following code
b).What are the contents of array **b** before the **for** loop to calculate **r**.

Hint: Do not multiply but cancel out common factors by computing GCDs. When we finally multiply the numbers, no number will become too large, because their product is known to be small.

```
#include <iostream>
using namespace std;
//gcd finds the gcd of x, y
unsigned int gcd( unsigned int x, unsigned int y) {
  unsigned int z;
  // GCD calculations using Euclid's theorem
  while ( x != 0 ) {
     z = __________;
     x = __________ % __________;
     y = ________________;
  }
  return y;
}

int main() {
       unsigned int a[10], b[10];
         for (int i =0; i<10; i++) {
                 cin>>a[i];
         }

         for (int i =0; i<10; i++) {
                 cin >>b[i];
```

```
        }
       // Cancel out common factors between every numerator denominator
pair
        for(int i=0; i<10; i++) {
                for(int j=0; j<10; j++){
                 int c = _________________________________;
                 b[i] = b[i]/_____________;
             a[j] = a[j]/___________;
                }
        }
      //For part b) give contents of array b at this position
        unsigned int r =1;
        for(int i=0; i<10; i++)
                r *= _____________;

        cout<<r<<endl;
}
```

Answer:

(a)

```
#include <iostream>
using namespace std;
//gcd finds the gcd of x, y
unsigned int gcd( unsigned int x, unsigned int y) {
  unsigned int z;
  // GCD calculations using Euclid's theorem
  while ( x != 0 ) {
     z = x; x = y % x;  y = z;
  }
  return y;
}

int main() {
```

Roll No ______________________________ Block No _____________

```
    unsigned int a[10], b[10];
      for (int i =0; i<10; i++) {
              cin>>a[i];
      }

      for (int i =0; i<10; i++) {
              cin >>b[i];
      }
     // Cancel out common factors between every numerator denominator pair
      for(int i=0; i<10; i++) {
              for(int j=0; j<10; j++){
               int c = gcd(b[i], a[j]);
                b[i] = b[i]/c;
            a[j] = a[j]/c;
              }
      }
      unsigned int r =1;
      for(int i=0; i<10; i++)
              r *= a[i];

      cout<<r<<endl;
}
```

(b) Since the result r is known to be an integer, each element of the denominator will vanish after cancelling out all the common factors. Hence, all the elements of array b are 1.

Evaluation instructions

- The last blank in (a) can also include b[i] as it's technically correct since all of b[i] are 1.
- 1+1+1+1, 3+3+3+3

Roll No ______________________________   Block No _____________

Q7). [15 Marks] Given below is a program which implements the famous sorting algorithm called Merge-Sort. It sorts an array of integers. The function **mergeSort** prints the array at the end of each function call. Write out the output due to these print statements.
Assume that the **merge(int[], int l, int m, int r)** function is given. Its explanation is given after the code.

```
#include <iostream>
using namespace std;

/* l is for left index, r is right index of the
   sub-array of arr to be sorted and n is the size of the array*/
void mergeSort(int arr[], int l, int r, int n)
{
   if (l < r)
   {

      int m = (l+r)/2;

      // Sort first and second halves
      mergeSort(arr, l, m, n);
      mergeSort(arr, m+1, r, n);

         // Here it merges sub-arrays in between (l,m) and (m + 1, r).
      merge(arr, l, m, r);

      //Printing the array
      for (int i = 0; i < n; ++i)
      {
        cout << arr[i];
        if(i != n - 1) cout << " ";
      }
      cout << "\n";
   }
}


int main()
{
   int n = 6;
   int a[] = {9,1,8,2,7,3};
   int i;
```

Roll No ________________________________ Block No ______________

```
    mergeSort(a, 0, n - 1, n);
}
```

Explanation for the merge function:
The merge function takes two sorted subarrays of the main array as input and modifies the array such that these two subarrays are merged into a single sorted subarray.
For example: Consider array A = {7, 1, 2, 6, 2, 4, 5}. Then merge(A, 1, 3, 6) considers the subarray in A from 1st index to 3rd index i.e. {1, 3, 6} and then takes subarray from 4th index to 6th index i.e. {2, 4, 5} and now as you can see these are two sorted subarrays of A, on merging them we obtain A = {7, 1, 2, 3, 4, 5, 6}. The underlined part is result of the merging of the of subarrays in between (1,3) and (4,6) indices.

Answer:

1 9 8 2 7 3
1 8 9 2 7 3
1 8 9 2 7 3
1 8 9 2 3 7
1 2 3 7 8 9

Evaluation instructions:
3 marks for each step

Total: 15 marks

Roll No ______________________________ Block No _____________

Q8). [16 Marks] Following code prints all permutations of a given string(all characters are unique in the string). Fill in the blanks.

E.g., “ABC”
First generate all permutations of “BC”. Then, insert a single “A” on all possible positions of the permutations of the “BC” string.

For permutations of “BC”, generate all permutations of “C” which is “C” itself. And now by adding B its permutations are “BC” and “CB”.

P(“C”) = {“C”}
P(“BC”) = {“BC”, “CB”} //insert B in front of C, insert B at end of C
P(“ABC”) = {“ABC”, “BAC”, “BCA”, //Insert A in possible places in “BC”
“ACB”, “CAB”, “CBA” //Insert A in possible places in “CB”
}
Note: Refer to the STL commands given at the top of the paper for your help.

E.g.,
string str = “abcdef”
str.substr(1,3) returns “bcd”
str.substr(2) returns “cdef”
str.substr(6) returns “”
str.length() returns 6

<u>**Code:**</u>

```
#include <iostream>
#include <vector>
using namespace std;
vector<string> all_permutations(string str){
   vector<string> ans;
   if (str.length() == 1){
      ans.push_back(___________); //base case
      return ans;
   }
   vector<string> permutations = all_permutations(str.substr(1));

   for (size_t i=0; i<permutations.size(); i++){

       //Inserting character in all the permutations
```

Roll No ________________________________ Block No ______________

```
        string current = permutations[i];
        for (int j=0; j<______________; j++){
            string new_permutation = current.substr(0, j);
            new_permutation += str[____________];
            new_permutation += current.substr(__________);//Single integer
            ans.push_back(new_permutation);
        }
    }
    return ans;
}

int main(){
    string str;
    cin>>str;
    vector<string> vec = all_permutations(str);
    for(size_t i=0; i<vec.size(); i++){
        cout<<vec[i]<<" ";
    }
}
```

Answer:
Blank 1 - str
Blank 2 - str.size() or str.length() or current.length()+1 or current.size()+1
Blank 3 - 0
Blank 4 - j

Evaluation instructions:
3+3+5+5
Total: 16 marks