**Q1.** What will be the output of the following code? Graphically represent the data structures after each loop.

[CLO 1] [2x5=10]

```
std::list<int>  L;
std::stack<int> S;
std::queue<int> Q;
for (int i=1; i<=10; i++) {
    if (i<=5)
        L.push_back(i);
    else
        L.push_front(i);
}
//(a) Draw the list after executing the previous loop

int j=0;
for (auto it = L.begin(); it != L.end(); ++it) {
    if (j%2 == 0)
        S.push(*it);
    else
        Q.push(*it);
    ++j;
}
//(b) Draw the stack and the queue after executing the previous loop

L.clear();
while (!S.empty()) {
    L.push_back(S.top());
    S.pop();
}
//(c) Draw the list and the stack after executing the previous loop

while (!Q.empty()) {
    L.push_back(Q.front());
    Q.pop();
}
//(d) Draw the list and the queue after executing the previous loop

//(e) Display the output of the following loop
while(!L.empty()) {
    std::cout<<L.front()<<" ";
    L.pop_front();
}
```

**Q2.** Given the following code: [CLO 1] [0.5+0.5+3=4]

```
int f(int n) {
  if (n < 0)
    return -1*n;
  else if (n<=1)
    return n;
  else
    return n + f(n-3) + f(n-5);
}
```

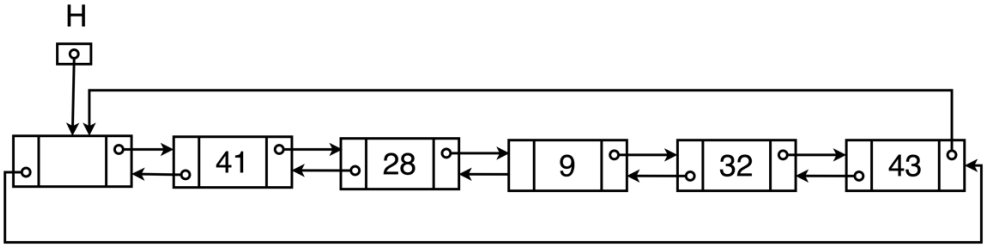
**a)** Identify the base case(s).

**b)** Identify the general case(s).

**c)** Draw the recursive tree for the function call `f(10)`.

**Q3.** Assume the following linked structures are created using nodes having three parts: a value, a pointer to the next node and a pointer to the previous node. Answer this question based on the linked structures given below:

[CLO 1] [1+2+3+2+2=10]




**a)** Write a single line of code to display the value 32.

**b)** Write a single line of code to check whether the values in the first node (41) and the last node (43) are equal.

**c)** Write code to display the values in the list in reverse order using a loop.

**d)** Write code to insert a new value 35 between the nodes with values 41 and 28. Do not use a loop.

**e)** Write code to delete the node with value 9. Do not use a loop.

**Q4.** Show all intermediate steps. No code is required for this question. [CLO 1] [4+2=6]

**a)** How would an empty AVL tree look like after inserting the following values in the given order?
41, 28, 9, 32, 43, 45, 73, 81, 96, 66.

**b)** How would the AVL in part (a) look like after removing the root node? If required, use successor node for removal.

**Q5.** How would an empty hash table of size 13 using open addressing and double hashing look like after inserting the following values? Assume that the second hash function returns the value: `k%(m-2)`, where `k` is the key and `m` is the hash table size. The values are 41, 28, 9, 32, 43, 45, 73, 81, 96, 66. Show all intermediate steps. No code is required for this question. [CLO 1] [6]

**Q6.** Sort the following values using heap sort. Assume you cannot use any additional array/space. 41, 28, 9, 32, 43, 45, 73. Show all intermediate steps. No code is required for this question. [CLO 1] [6]

**Q7.** [CLO 1] [6+2=8]

**a)** What will be the output of the following code segment?

```
char c_arr[] = {'D','A','T','A','S','T','R'};
std::map<char, std::pair<char, int> > m;
for (int i=0;i<6;i++)
    m[c_arr[i]] = {c_arr[i+1], i+1};

for (auto i=m.begin(); i!=m.end(); ++i)
    std::cout << i->first << "-" << i->second.first << i->second.second << std::endl;
```

**b)** Graphically draw the data structures in part (a). Assume that the map is unbalanced.

**DSA FORMULA SHEET**

| | | |
|---|---|---|
| 1. | Total Nodes in full map/tree: | $n=2^h-1$ |
| 2. | Height of full map/tree: | $h=\log_2(n+1)$ |
| 3. | Left child: | i*2+1 |
| 4. | Right child: | i*2+2 |
| 5. | Parent: | (i-1)/2 |
| 6. | First leaf: | (n/2) |
| 7. | Range of chaining: | [n/10,n/5] |
| 8. | Range of open addressing: | [n*1.3,n*1.7] |
| 9. | Linear Hashing: | (h + i)%size |
| 10. | Quadratic Hashing: | (h + i*i)%size |
| 11. | Double Hashing: | (h1 + i*(h2+1))%size |