Assignment 1
(Week 1 – Lab A & Lab B)

Q1. Considering that an integer variable, a float variable, a double variable, a character variable, and a pointer variable need 4, 4, 8, 1, and 8 bytes memory space respectively, what will be the output of following C++ programs.

(a)

```
#include <iostream>
using namespace std;
class abc
{
   int x;
   double y;
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   cout<<"\nSize of abc is : "<<sizeof(abc);
   return 0;
}
```

(b)

```
#include <iostream>
using namespace std;
class abc
{
   int x;
   double y;
   int z;
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   return 0;
}
```

(c)

```
#include <iostream>
using namespace std;
class abc
{
   int x;
   double y;
   int z;
   int a;
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   return 0;
}
```

(d)

```
#include <iostream>
using namespace std;
class abc
{
   float x;
   char y;
   int z;
   double a;
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   return 0;
}
```

(e)

```
#include <iostream>
using namespace std;
class abc
{
   char x[5];
   double y;
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   return 0;
}
```

(f)

```
#include <iostream>
using namespace std;
class abc
{
   char x[5];
   float y[3];
};
int main()
{
   abc *o1 = new abc, o2;
   cout<<"\nSize of o1 : "<<sizeof(o1);
   cout<<"\nSize of o2 : "<<sizeof(o2);
   return 0;
}
```

Q2. Analyze the correctness and output of following programs

(a)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        float *a;
        a = (float *)malloc(sizeof(int));
        a[0] = 4.5;
        cout<<a[0];
        return 0;
}
```

(b)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a;
        a = (int *)malloc(sizeof(float));
        a[0] = 5;
        cout<<a[0];
        return 0;
}
```

(c)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a, *b;
        a = (int *)malloc(sizeof(int));
        b = (int *)malloc(5*sizeof(int));
        cout<<sizeof(a)<< sizeof(b);
        return 0;
}
```

(d)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a;
        a[0] = (int *)malloc(sizeof(int));
        a[0] = 5;
        cout<<a[0];
        return 0;
}
```

(e)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a[5];
        a[0] = (int *)malloc(sizeof(int));
        a[0][0] = 5;
        cout<<a[0][0];
        return 0;
}
```

(f)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        struct node{int a[10];};
        struct node *n;
        n = (struct node *)malloc(sizeof(struct node));
        cout<<sizeof(n);
        return 0;
}
```

(g)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a[5];
        a[0] = (int *)malloc(2*sizeof(int));
        a[0][1] = 5;
        cout<<a[0][1];
        return 0;
}
```

(h)

```
#include <iostream>
#include <malloc.h>
using namespace std;
int main() {
        int *a = (int *)malloc(5*sizeof(int));
        a[0] = 1; a[1] = 2; a[2] = 3; a[3] = 4; a[4] = 5;
         delete(a);
         cout<<a[0]<<a[1]<<a[2]<<a[3]<<a[4];
         return 0;
}
```

Q3. A dynamically created array stores following integer elements (odd and even integers)

2	8	3	6	7	9	5	4

It is desired to print/display the elements of this array in such manner that it first prints all the even elements then it prints all the odd elements. In above example, the displayed elements are as follows:

2 8 6 4 3 7 9 5

Write a program in C/C++ to create the dynamic array of user inputted length (n), assign values at different indices of the array, and as presented in above example, display the elements of this array.

(**Note:** don't enter the elements manually, rather use following statement in loop to randomly assign elements (in range between 0 and 99) in the array: A[i] = rand()%100, where A is an array)

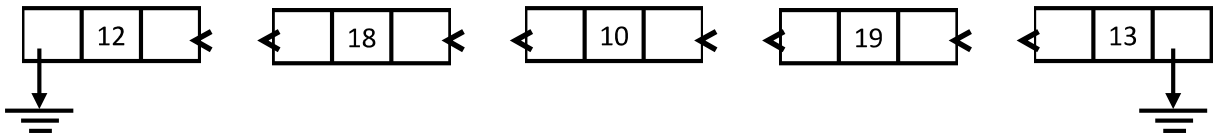
Q4. Write a program in C/C++ to create a Singly Linked List to store an integer element in each of the n nodes of the Singly Linked List and store/assign elements to all the n nodes. Further in your program, delete the nodes of this Linked List as per the following rules: always delete that node having smallest integer in the Singly Linked List updated after deletion. Before deletion, display the element stored in the node.

(**Note:** don't enter the elements manually, rather use following statement randomly assign element in the node: ptr → info = rand()%100, where ptr is a node pointer)

Q5. Write a program in C/C++ to create a Circular Singly Linked List to store an integer element in each of the n nodes of the Circular Singly Linked List and store/assign elements to all the n nodes. Further in your program, delete the nodes of this Circular Linked List as per the following rules: always delete that node having smallest integer in the Circular Singly Linked List updated after deletion.

(**Note:** don't enter the elements manually, rather use following statement randomly assign element in the node: ptr → info = rand()%100, where ptr is a node pointer)

Q6. Following example depicts a Doubly Linked List (nodes and integer elements stored in the nodes)



It is desired to traverse the DLL in such a manner that it displays following pattern as the output:

10

18 10 19

12 18 10 19 13

Write a program in C/C++ to create a Doubly Linked List to store an integer element in each of the n nodes of the Doubly Linked List and store/assign elements to all the n nodes. (**Note:** don't enter the elements manually, rather use following statement randomly assign element in the node: ptr → info = rand()%100, where ptr is a node pointer). Considering that you are allowed to use the starting pointer only once, traverse the DLL to display the elements in above depicted manner.