Ans to the Question Number – 1
factorial

Recursive Method:

```
#include<bits/stdc++.h>
using namespace std;

int factorial(int fact, int n)
{
   if (n == 0)
   {
      return fact = 1;
   }
   else
   {
      return fact = n * factorial(fact, n - 1);
   }
}

int main()
{
   int n, fact;
   cin >> n;
   fact = 1;
   cout << factorial(fact, n) << endl;
   return 0;
}
```

Non-Recursive Method:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int i, fact = 1, n;
   cin >> n;
```

```
    for (i = 1; i <= n; i++)
    {
        fact = fact * i;
    }
    cout << fact << endl;
    return 0;
}
```

Ans to the Question Number – 2
nth term F(n) of the Fibonacci sequence

Recursive Method:

```
#include<bits/stdc++.h>
using namespace std;

int fibonacci(int n)
{
    if (n <= 1)
        return n;
    else
        return fibonacci(n - 1) + fibonacci(n - 2);
}

int main()
{
    int n;
    cout << "Enter a number : ";
    cin >> n;
    int ans = fibonacci(n);

    cout << ans << endl;

    return 0;
}
```

Non-Recursive Method:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int first = 0, second = 1, cnt = 0, fibonacci, n;

   cout << "Enter range : ";
   cin >> n;

   while (cnt < n)
   {
      if (cnt <= 1)
         fibonacci = cnt;
      else
      {
         fibonacci = first + second;
         first = second;
         second = fibonacci;
      }
      cout << fibonacci << endl;
      cnt++;
   }
   return 0;
}
```

Ans to the Question Number – 3
move n disks for Tower of Hanoi problem.

```
#include<bits/stdc++.h>
using namespace std;

void Tower_of_Hanoi(int n, char start, char ax, char end)
{
   if (n == 1)
   {
       cout << start << "->" << end << endl;
       return;
   }
   else
   {
       Tower_of_Hanoi(n - 1, start, end, ax);
       cout << start << "->" << end << endl;

       Tower_of_Hanoi(n - 1, ax, start, end);
       return;
   }
}

int main()
{
   int n;
   cout << "Enter number: ";
   cin >> n;
   Tower_of_Hanoi(n, 'A', 'B', 'C');

   return 0;
}
```

Ans to the Question Number – 4 value from Ackerman function.

```
#include<bits/stdc++.h>
using namespace std;

int ackerman(int x, int y)
{
    if (x == 0)
        return y + 1;
    else if (x != 0 && y == 0)
        return ackerman(x - 1, 1);
    else
```

```
        return ackerman(x - 1, ackerman(x, y - 1));
}

int main ()
{
    int x, y;
    cout << "Enter two number: ";
    cin >> x >> y;
    cout << ackerman(x, y) << endl;
    return 0;
}
```

Ans to the Question Number – 5
insert and delete operations of a circular queue.

```
#include<bits/stdc++.h>
using namespace std;

int queue[6], beg = 0, end = 0, item;
int N = sizeof(queue) / sizeof(queue [0]) - 1;

void Insert ()
{
    if ((beg == 1 && end == N) || (beg == end + 1))
    {
        cout << "\nOverFlow" << endl;
        return;
    }
    if (beg == 0)
    {
        beg = 1;
        end = 1;
    }
    else if (end == N)
        end = 1;
    else
    {
       end = end + 1;
    }
    cout << "\nEnter number: ";
    cin >> item;
    queue[end] = item;
    return;
}

void Delete ()
{
    if (beg == 0)
    {
       cout << "\nUnderFlow" << endl;
       return;
    }
    item = queue[beg];
    if (beg == end)
    {
       beg = 0;
       end = 0;
    }
    else if (beg == N)
       beg = 1;
    else
       beg = beg + 1;
    return;
}

void Show ()
{
    if (beg == 0)
    {
       cout << "\nQueue is Empty" << endl;
    }
    else
    {
       cout << "\nShow Queue" << endl;
```

```
        if (beg <= end)
        {
           for (int i = beg; i <= end; i++)
           {
              cout << "queue [" << i << "] = " << queue[i] << endl;
           }
        }
        else if (end < beg)
        {
           for (int i = beg; i <= N; i++)
           {
              cout << "queue [" << i << "] = " << queue[i] << endl;
           }
           for (int i = 1; i <= end; i++)
           {
              cout << "queue [" << i << "] = " << queue[i] << endl;
           }
        }

     }
  }

  int main ()
  {

     int choice;

     while (1)
     {
        cout << "\n1. Insert" << endl;
        cout << "2. Delete" << endl;
        cout << "3. Show" << endl;
        cout << "4. Exit" << endl;
        cout << "\nEnter your choice: ";
        cin >> choice;

        if (choice == 4)
           break;
        switch (choice)
        {
        case 1:
        {
           Insert ();
           break;
        }
        case 2:
        {
           Delete ();
           break;
        }
        case 3:
        {
           Show ();
           break;
        }
        }
    }
    return 0;

}
```

Ans to the Question Number – 6

Ans to the Question Number – 7
insert and delete operations of a priority queue using array.

```
#include<bits/stdc++.h>
using namespace std;
```

```
int queue [6][6], Item;

int N = sizeof(queue) / sizeof(queue [0]) - 1;

struct Priority
{
   int front = 0;
   int rear = 0;
};
struct Priority number [5];

void Insert ()
{
   int prio_num;
   cout << "Enter priority number: ";
   cin >> prio_num;

   if ((number[prio_num].front == 1 && number[prio_num].rear ==
N) || (number[prio_num].front == number[prio_num].rear + 1))
   {
      cout << "\nOverFlow" << endl;
      return;
   }

   if (number[prio_num].front == 0)
   {
      number[prio_num].front = 1;
      number[prio_num].rear = 1;
   }

   else if (number[prio_num].rear == N)
      number[prio_num].rear = 1;


   else
   {
      number[prio_num].rear = number[prio_num].rear + 1;
   }
```

```
    cout << "\nEnter number: ";
    cin >> Item;
    queue[prio_num][number[prio_num].rear] = Item;

    return;
}

void Delete ()
{
    for (int i = 1; i <= N; i++)
    {
        if (number[i].front == 0)
        {
            if (i == N)
            {
                cout << "\nUnderFlow" << endl;
                return;
            }
            else
                continue;
        }

        Item = queue[i][number[i].front];

        if (number[i].front == number[i].rear)
        {
            number[i].front = 0;
            number[i].rear  = 0;
        }

        else if (number[i].front == N)
            number[i].front = 1;

        else
            number[i].front = number[i].front + 1;
        return;
    }
```

```
}

void Show ()
{
   int priority_num;
   cout << "Enter priority number: ";
   cin >> priority_num;

   if (number[priority_num].front == 0)
   {
      cout << "\nQueue is Empty" << endl;
   }

   else
   {
      cout << "\nQueue Show!!" << endl;

      if (number[priority_num].front <= number[priority_num].rear)
      {
         for (int i = number[priority_num].front ; i <=
number[priority_num].rear; i++)
         {
            cout << "queue[" << priority_num << "][" << i << "] = " <<
queue[priority_num][i] << endl;
         }
      }

      else if (number[priority_num].rear <
number[priority_num].front)
      {

         for (int i = number[priority_num].front; i <= N; i++)
         {
            cout << "queue[" << priority_num << "][" << i << "] = " <<
queue[priority_num][i] << endl;
         }

         for (int i = 1; i <= number[priority_num].rear; i++)
```

```
        {
            cout << "queue[" << priority_num << "][" << i << "] = " <<
queue[priority_num][i] << endl;
        }
      }

   }
}

int main ()
{
   int choice;
   while (1)
   {
      cout << "\n1. Insert" << endl;
      cout << "2. Delete" << endl;
      cout << "3. Show" << endl;
      cout << "4. Exit" << endl;
      cout << "\nEnter your choice: ";
      cin >> choice;

      if (choice == 4)
         break;

      switch (choice)
      {
      case 1:
      {
         Insert ();
         break;
      }

      case 2:
      {
         Delete ();
         break;
      }
```

```
        case 3:
        {
           Show ();
           break;
        }
        }
    }
    return 0;

}
```

Ans to the Question Number – 8
create a Linked List of n elements and then display the list.

```
#include<bits/stdc++.h>
using namespace std;
#define NULL 0

struct linked_list
{
    int num;
    struct linked_list *next;
};
typedef struct linked_list node;

int main()
{
    int n, i, item;
    node *start, *ptr;

    start = (node *) malloc(sizeof(node));
    ptr = start;

    printf("How many elements: ");
    scanf("%d", &n);

    for (i = 1; i <= n; i++)
```

```
  {
    printf("Enter a number: ");
    scanf("%d", &ptr->num);
    if (i != n)
    {
      ptr->next = (node *)malloc(sizeof(node));
      ptr = ptr->next;
    }
  }
  ptr->next = NULL;

  printf("\nElements in the link list are: \n");
  ptr = start;
  while (ptr != NULL)
  {
    printf("%d\n", ptr);
    printf("%d\n", ptr->num);
    ptr = ptr->next;
  }

  return 0;
}
```

Ans to the Question Number – 9
create a Linked List of n elements and then search an element from the list.

```
#include<bits/stdc++.h>
using namespace std;
#define NULL 0

struct linked_list
{
  int num;
  struct linked_list *next;
};
```

```
typedef struct linked_list node;

int main ()
{
   int n, i, item, cnt = 0;
   node *start, *ptr;

   start = (node *) malloc(sizeof(node));
   ptr = start;

   printf("How many elements: ");
   scanf("%d", &n);

   for (i = 1; i <= n; i++)
   {
      printf("input a number: ");
      scanf("%d", &ptr->num);
      if (i != n)
      {
         ptr->next = (node *)malloc(sizeof(node));
         ptr = ptr->next;
      }
   }
   ptr->next = NULL;

   cout << "Enter a number you want to search: ";
   cin >> item;

   ptr = start;
   while (ptr != NULL)
   {
      if (item == ptr->num)
      {
         cout << "Location = " << ptr << endl;
         cnt = 1;
         break;
      }
      else
```

```
        ptr = ptr->next;

    }

    if (cnt == 0)
       cout << "Location = " << NULL << endl;

    return 0;
}
```

Ans to the Question Number – 10
create a Linked List of n elements and then insert an element to the list.

```
#include<bits/stdc++.h>
using namespace std;
#define NULL 0

struct linked_list
{
   int num;
   struct linked_list *next;
};
typedef struct linked_list node;

node *start, *ptr, *Loc, *New, *save;

Create ()
{
   int Number, i;
   start = (node *) malloc(sizeof(node));
   ptr = start;

   printf("How many elements: ");
   scanf("%d", &Number);

   for (i = 1; i <= Number; i++)
   {
        printf("input a number: ");
        scanf("%d", &ptr->num);
        if (i != Number)
        {
            ptr->next = (node *)malloc(sizeof(node));
            ptr = ptr->next;
        }
    }
    ptr->next = NULL;

}


node *Find_Location(int item)
{
    ptr = start;
    if (start == NULL)
    {
        return NULL;
    }
    if (item < ptr->num)
    {
        return NULL;
    }
    save = start;

    while (ptr != NULL)
    {
        if (item < ptr->num)
        {
            return save;
        }
        save = ptr;
        ptr = ptr->next;

    }
    return save;
```

```
}

Ins_Location(node *Loc, int item)
{

   New = (node *) malloc(sizeof(node));
   New->num = item;

   if (Loc == NULL)
   {
      New->next = start;
      start = New;
   }
   else
   {
      New->next = Loc->next;
      Loc->next = New;
   }

}
int main ()
{
   int n, i, item;

   Create ();

   cout << "Enter a number you want to insert: ";
   cin >> item;

   Loc = Find_Location(item);

   Ins_Location(Loc, item);

   printf("\nElements in the link list are: \n");
   ptr = start;
   while (ptr != NULL)
   {
      printf("%d\n", ptr->num);
```

```
        ptr = ptr->next;
    }

    return 0;
}
```

Ans to the Question Number – 11 create a Linked List of n elements and then delete an element from the list.

```
#include<bits/stdc++.h>
using namespace std;
#define NULL 0

struct linked_list
{
    int num;
    struct linked_list *next;
};
typedef struct linked_list node;
node *start, *ptr, *Loc, *LocPrev, *New, *save;

Create ()
{
    int Number, i;
    start = (node *) malloc(sizeof(node));
    ptr = start;

    printf("How many elements? : ");
    scanf("%d", &Number);

    for (i = 1; i <= Number; i++)
    {
        printf("input a number: ");
        scanf("%d", &ptr->num);
        if (i != Number)
        {
            ptr->next = (node *)malloc(sizeof(node));
            ptr = ptr->next;
        }
    }
    ptr->next = NULL;

}
node *FindLoc(int item)
{
    ptr = start;
    if (start == NULL)
    {
        LocPrev = NULL;
        return NULL;
    }
    if (ptr->num == item)
    {
        LocPrev = NULL;
        return start;
    }

    save = start;

    while (ptr != NULL)
    {
        if (ptr->num == item)
        {
            LocPrev = save;
            return ptr;
        }

        save = ptr;
        ptr = ptr->next;

    }
    return NULL;
```

```
}

Delete (node *Loc, node *LocPrev, int item)
{
   ptr = start;
   if (Loc == NULL)
   {
      cout << "Item is not in list" << endl;

   }

   else if (LocPrev == NULL)
   {
      start = ptr->next;
   }

   else
      LocPrev->next = Loc->next;
}

int main ()
{
   int n, i, item;

   Create ();

   cout << "Enter a number you want to delete: ";
   cin >> item;

   Loc = FindLoc(item);

   Delete (Loc, LocPrev, item);

   printf("\nElements in the link list are: \n");
   ptr = start;
   while (ptr != NULL)
   {
      printf("%d\n", ptr->num);
```

```
        ptr = ptr->next;
    }

    return 0;
}
```

Ans to the Question Number – 12
create a Circular Header Linked List of n elements and then display the list.

```
#include<bits/stdc++.h>
using namespace std;
#define NULL 0

struct linked_list
{
    int num;
    struct linked_list *next;
};
typedef struct linked_list node;

int main ()
{
    int n, i;
    node *start, *ptr, *header;

    start = (node *) malloc(sizeof(node));
    header = start;

    ptr = start;
    ptr->next = (node *) malloc(sizeof(node));

    printf("How many elements? ");
    scanf("%d", &n);

    for (i = 1; i <= n; i++)
    {
       printf("Insert a number: ");
       scanf("%d", &ptr->num);
       if (i != n)
       {
          ptr->next = (node *)malloc(sizeof(node));
          ptr = ptr->next;
       }
    }
    ptr->next = header;

    printf("\nElements in the link list are: \n");

    ptr = header;

    do
    {
       cout << ptr->num << endl;
       ptr = ptr->next;
    }
    while (ptr != header);

    return 0;
}
```

Ans to the Question Number – 13

Ans to the Question Number – 14 find the 100!(Factorial).

```
#include<bits/stdc++.h>
using namespace std;
int main ()
{
    int n;
    cin >> n;

    int q = 2;
    int arr[100000] = {0};
    arr[0] = 1;
    int len = 1;
    int x = 0;
    int num = 0;
    while (q <= n)
    {
       x = 0;
       num = 0;
       while (x < len)
       {
          arr[x] = arr[x] * q;
          arr[x] = arr[x] + num;
          num = arr[x] / 10;
          arr[x] = arr[x] % 10;
          x++;
       }
       while (num != 0)
       {
          arr[len] = num % 10;
          num = num / 10;
          len++;
       }
       q++;
    }
    len--;
    while (len >= 0)
    {
       cout << arr[len];
       len = len - 1;
    }
}
```

Ans to the Question Number – 15
the value of the nth Fibonacci number F(n) where F(n) = F(n–1) + F(n-2) and F(1) = F(2) = 1 and (n <= 500).

```
#include<bits/stdc++.h>
using namespace std;
#define optimize()          \
ios_base::sync_with_stdio(0); \
cin.tie(0);                 \
cout.tie(0);
const int maxn = 2e5+10;

#define vl vector<long long int>
#define vi vector<int>
#define pb push_back
#define all(x) (x).begin(), (x).end()

#define ll long long int
#define ld long double

#define fr(i,n) for (ll i=0;i<n;i++)
#define fr1(i,n) for(ll i=1;i<=n;i++)

#define endl '\n'

string add(string num1,string num2)
{
   string s1 = num1;
   string s2 = num2;

   reverse(s1.begin(),s1.end());
   reverse(s2.begin(),s2.end());

   if(s1.length()>s2.length()) swap(s1,s2);
```

```
    int carry = 0;
    string ans;
    //9999 porjonto add
    for(int i=0; i<s1.length(); i++)
    {
        int x=(s2[i]-'0')+(s1[i]-'0')+carry;
        ans+=x%10+'0';
        carry=x/10;
    }
    //9999 er por er add
    for(int i=s1.length(); i<s2.length(); i++)
    {
        int x=(s2[i]-'0')+carry;
        ans+=x%10+'0';
        carry=x/10;
    }
    if(carry) ans+=carry+'0';

    reverse(ans.begin(),ans.end());

    //cout << ans << endl;
    return ans;
}

string fibonacci(int a)
{
    string F[a+10];
    F[0] = "0", F[1] = "1";
    for(ll i=2; i<=a; i++)
    {
        F[i] = add(F[i-1],F[i-2]);
    }
    cout << endl;

    return F[a];
}
```

```
int main()
{
   ///Peace be with you.
   optimize();

   ll n;
   cout << "Enter the Index Number of FIBONACCI Sequence : ";
   cin >> n;
   cout << fibonacci(n) << endl;

   return 0;
}
```