Frequent Asked Questions

→ If we have a 4 dijit number X = ABCD

```
A = X/1000 ;
B = (X - (A*1000)) / 100 ;
C = (X - (A*1000) - (B*100))/10 ;
D = (X - (A*1000) - (B*100) - (C*10)) ;
```

→ Read a sequence of nb, stop when we enter a negative nb

```
int X ;
while (1)
{ cout << " Enter X : " ;
  cin >> X ;
  if (X < 0)
     break ;
}
```

→ Indicate if N is a prime nb

```
int N, c = 0, i ;
cout << " Enter N " ;
cin >> N ;
for ( i = 2 ; i < N ; i++)
{ if (N % i == 0)
     c = 1 ;
     break ;
}
if ( c == 0 )
 cout << N << " is a prime nb " ;
else
 cout << N << " is not a prime nb " ;
```

→ Sum of factorial From 1 to N

```
int N , i, j ;
double S = 0 ; F ;
For (i = 1; i <= N; i++ )
{ F= 1;
  for( j=1; j<= i ; j++ )
        F = F * j ;
  S = S + F;
}
```

→ maximum of Sequence of N nb

```
{ float x, nb1 , max;
  int i;
  cout << " Enter value 1 : " ;
  cin >> nb1 ;
  max = nb1;
  for (i = 2 ; i <= N ; i++)
  { cout << " Entervalue " << i << " : " ;
    cin >> x ;
    if ( x > max )
     max = x ;
  }
 cout << "maximum = " << max;
}
```

→ Array

```
// Assign the array with values
for (i=0 ; i<n ; i++)        // n: dimension of the array
{ cout << "Enter the value of A[" << i << "]";
  cin >> A[i];
}
```

one_dim

```
for (i=0 ; i<R ; i++)        // R: dimension of row
{ for(j=0; j<C ; j++)        // C: column
    cout << "Enter the value of A[" << i << "][" << j << "]";
    cin >> A[i][j];
}
```

two_dim

```
// Display the Array
for (i=0; i<n ; i++)
    cout << A[i] << "\t";
```

one_dim

```
for (i=0; i<r; i++)
{ for (j=0; j<c; j++)
      cout << A[i][j] << "\t";
  cout << endl;
}
```

two_dim

→ Sorting a one dimension array of dimension N

```
int T[N] , i, j, temp;
for(i=0 ; i<N; i++)
{ for(j=i+1 ; j<N ; j++)
    if( T[i]>T[j])
      { temp = T[i];
        T[i] = T[j];
        T[j] = temp;
      }
}
```

→ Multiplication of 2 matrix ($M_{R \times C} / M_{C \times P} \rightarrow M$

```
int R, C, P, i, j, K;
cout << " Enter the nb of rows of A : ";
cin >> R;
cout << " Enter the nb of columns of A and rows of B : ";
cin >> C;
cout << " Enter the nb of columns of B : ";
cin >> P;
int A[R][C], B[C][P], MULT[R][P];
// Assign A
// Assign B
// Initializing elements of MULT to 0
for(i=0; i<R; i++)
{ For(j=0; j<P; j++)
    MULT[i][j]=0;
}
// Multiplying A and B and sorting results in MULT
for(i=0; i<R; i++)
{   for(j=0; j<P; j++)
        for(K=0; K<C; K++)
            MULT[i][j] = MULT[i][j] + (A[i][K] * B[K][j]);
}

// Display MULT
for(i=0; i<R; i++)
{ (j=0; j<P, j++)
    cout << MULT[i][j] << "\t";
cout << endl;
}
```

→ Transfer a two_dim array to one_dim array.

```
M[R][C] , V[R*C]
int K=0
for (i=0 ; i<R ; i++)
    for (j=0 ; j<C ; j++)
        V[K++] = M[i][j] ;
```

→ Function that calculate U_n , U_0, U_1

$U_n = 2 * U_{n-1} + U_{n-2} + 3 \quad (n \geq 2)$

```
int U (int n, int U0, int U1)
{ int U,i, A=U1 , B=U0;
  if ( n==0)
     U= U0 ;
        else { if (n==1)
                  U = U1;
                     else { for (i=2; i<=n ; i++)
                              U= 2 * A + B + 3 ;
                               B = A;
                               A = U;
                            }
                  }
   return U;
}

int S ( int n; int U0, int U1)
{ int i, S=0 ;
   for (i=0; i<=n; i++)
     S= S+ U(n, U0, U1) ;
  return S;
}
```