**Lab 02 Tasks**

**# Name:** Areeba Zubair

**# Roll Number:** 24k-0596

**All Libraries:**

#include "iostream"

#include <string>

#include <cstring>

using namespace std;

**Task 01:**

int main()

{   int size;

    int find = 0;

    cout<< "Enter the size of array: ";

    cin >> size;

    int\* ptr = new int[size];

    for(int i=0; i<size; i++)

    {

        cout<< "Element "<<i+1<<": ";

        cin>>ptr[i];

        find+=ptr[i];

    }

    cout<<endl;

    cout<<"Sum = "<<find<<endl;

    cout<<"Average = "<<float(find)/size<<endl;

    find = 0;

    for(int i=0; i<size; i++)

    {  if(ptr[i] > find)

       find = ptr[i];

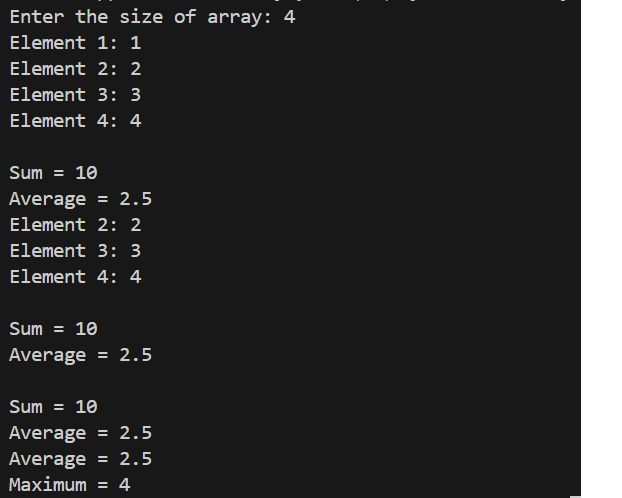
    }

    cout<<"Maximum = "<<find<<endl;

    delete[] ptr;

}

**Output:**



**Task 02:**

int\*\* Allocation(int r, int c)

{   int\*\* ptr = new int\*[r];

    for(int i=0; i<r; i++)

    {  ptr[i] = new int[c];

    }

    return ptr;

}

void Input(int\*\* ptr, int r, int c)

{   for(int i=0; i<r; i++)

    {   for(int j=0; j<c; j++)

        {  cout<<"Element ["<<i<<"]["<<j<<"]: ";

           cin>>ptr[i][j];

        }

    }

}

int\*\* Addition(int\*\* ptr1, int\*\* ptr2, int r, int c)

{  int \*\*add = new int\*[r];

   for(int i=0; i<r; i++)

    {  add[i] = new int[c];

    }

    for(int i=0; i<r; i++)

    {   for(int j=0; j<c; j++)

        {   add[i][j] = ptr1[i][j] + ptr2[i][j];

        }

    }

    return add;

}

int\*\* Subtraction(int\*\* ptr1, int\*\* ptr2, int r, int c)

{  int \*\*subtract = new int\*[r];

   for(int i=0; i<r; i++)

    {   subtract[i] = new int[c];

    }

    for(int i=0; i<r; i++)

    {   for(int j=0; j<c; j++)

        {   subtract[i][j] = ptr2[i][j] - ptr1[i][j];

        }

    }

    return subtract;

}

int\*\* Multiplication(int\*\* ptr1, int\*\* ptr2, int r1, int c1, int r2, int c2)

{   int a, b = 0;

    int \*\*multiply = new int\*[r1];

    for(int i=0; i<r1; i++)

    {  multiply[i] = new int[c2];

    }

    for(int i=0; i<r1; i++)

    {   for(int j=0; j<c2; j++)

        {   for(int k=0; k<r2; k++)

            {   a = ptr1[i][k]\*ptr2[k][j];

                b+=a;

            }

            multiply[i][j] = b;

            b=0;

        }

    }

    return multiply;

}

void free(int\*\* ptr,int r)

{   for(int i=0; i<r; i++)

    {  delete[] ptr[i];

    }

    delete[] ptr;

}

int main()

{   int row1, cols1, row2, cols2;

    cout<<"Enter rows of matrix 01: ";

    cin>>row1;

    cout<<"Enter columns of matrix 01: ";

    cin>>cols1;

    cout<<"Enter rows of matrix 02: ";

    cin>>row2;

    cout<<"Enter columns of matrix 02: ";

    cin>>cols2;

    //Allocation

    int \*\*ptr1 = Allocation(row1, cols1);

    int \*\*ptr2 = Allocation(row2, cols2);

    //Inputs

    cout<<"Matrix 1\n";

    Input(ptr1, row1, cols1);

    cout<<"Matrix 2\n";

    Input(ptr2, row2, cols2);

    //Addition Operation

    if(row1 == row2 && cols1 == cols2)

    {   int \*\*addition = Addition(ptr1,ptr2,row1,cols1);

        cout<<"Addition\n";

        for(int i=0; i<row1; i++)

        {   cout<<"|  ";

            for(int j=0; j<cols1; j++)

            {   cout<<\*(\*(addition+i)+j)<<"  ";

            }

            cout<<"|";

            cout<<endl;

        }

        free(addition,row1);

    }else{

        cout<<"Addition is not possible\n";

    }

    //Subtraction Operation

    if(row1 == row2 && cols1 == cols2)

    {   int \*\*subtraction = Subtraction(ptr1,ptr2,row1,cols1);

        cout<<"Subtraction\n";

        for(int i=0; i<row1; i++)

        {   cout<<"|  ";

            for(int j=0; j<cols1; j++)

            {   cout<<\*(\*(subtraction+i)+j)<<"  ";

            }

            cout<<"|";

            cout<<endl;

        }

        free(subtraction,row1);

    }else{

        cout<<"Subtraction is not possible\n";

    }

    //Multiplication Operation

    if(cols1 == row2)

    {   int\*\* multiply = Multiplication(ptr1,ptr2,row1,cols1,row2,cols2);

        cout<<"Multiplication\n";

        for(int i=0; i<row1; i++)

        {   cout<<"|  ";

            for(int j=0; j<cols2; j++)

            {   cout<<\*(\*(multiply+i)+j)<<"  ";

            }

            cout<<"|";

            cout<<endl;

        }

        free(multiply,row1);

    }else{

        cout<<"Multiplication is not possible\n";

    }

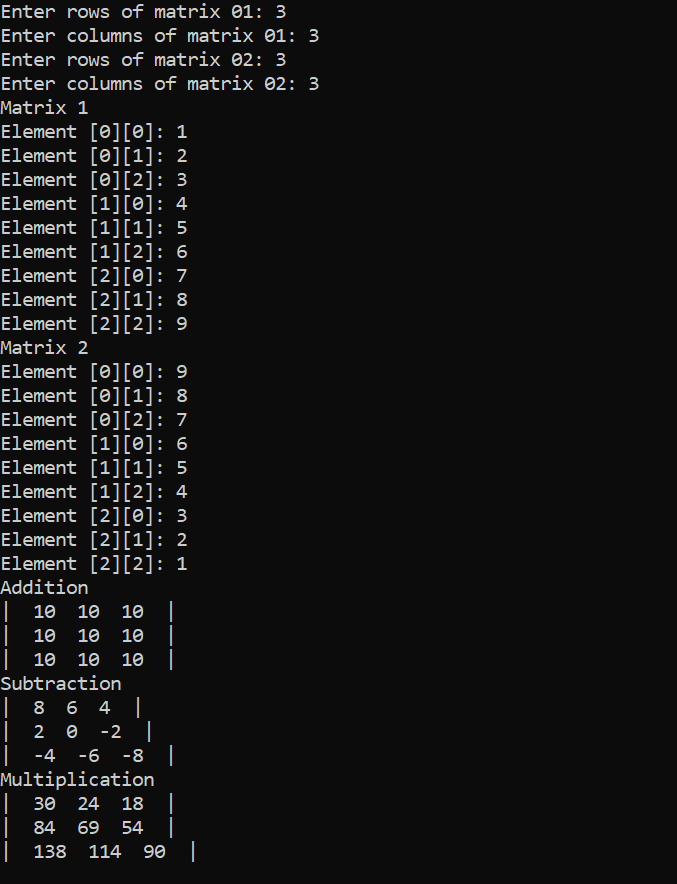
    //Deallocation

    free(ptr1,row1);

    free(ptr2,row2);

}

**Output:**



**Task 03:**

struct Employee{

   int employeeId;

   string name;

   string department;

   int salary;

};

void Input(Employee\* emp, int num)

{   for(int i=0; i<num; i++)

    {   cout<<"Employee "<<i+1<<endl;

        cout<<"Employee Id: ";

        cin>>emp[i].employeeId;

        cout<<"Name: ";

        cin>>emp[i].name;

        cout<<"Department: ";

        cin>>emp[i].department;

        cout<<"Salary: ";

        cin>>emp[i].salary;

    }

    cout<<endl;

}

void Display(Employee\* emp, int num)

{   cout<<"------------------------------------------"<<endl;

    for(int i=0; i<num; i++)

    {   cout<<"Employee "<<i+1<<endl;

        cout<<"Employee Id: "<<emp[i].employeeId<<endl;

        cout<<"Name: "<<emp[i].name<<endl;

        cout<<"Department: "<<emp[i].department<<endl;

        cout<<"Salary: "<<emp[i].salary<<endl;

        cout<<"------------------------------------------"<<endl;

    }

    cout<<endl;

}

void Searching(Employee\* emp, int num)

{   int id, found=0;

    cout<<"Enter ID: ";

    cin>>id;

    for(int i=0; i<num; i++)

    {   if(id == emp[i].employeeId)

        { found++;

          cout<<"Employee "<<i+1<<endl;

          cout<<"Employee Id: "<<emp[i].employeeId<<endl;

          cout<<"Name: "<<emp[i].name<<endl;

          cout<<"Department: "<<emp[i].department<<endl;

          cout<<"Salary: "<<emp[i].salary<<endl;

        }

    }

    if(!found)

    cout<<"This ID: "<<id<<" doesn't exit\n";

}

int main()

{   int numOfemp;

    cout<<"Number of Employee: ";

    cin>>numOfemp;

    Employee\* emp = new Employee[numOfemp];

    Input(emp,numOfemp);

    Display(emp,numOfemp);

    string wanna;

    do{

    cout<<"Do you wanna search about any employee(y/n): ";

    cin>>wanna;

    if(wanna == "y")

    {  Searching(emp,numOfemp);

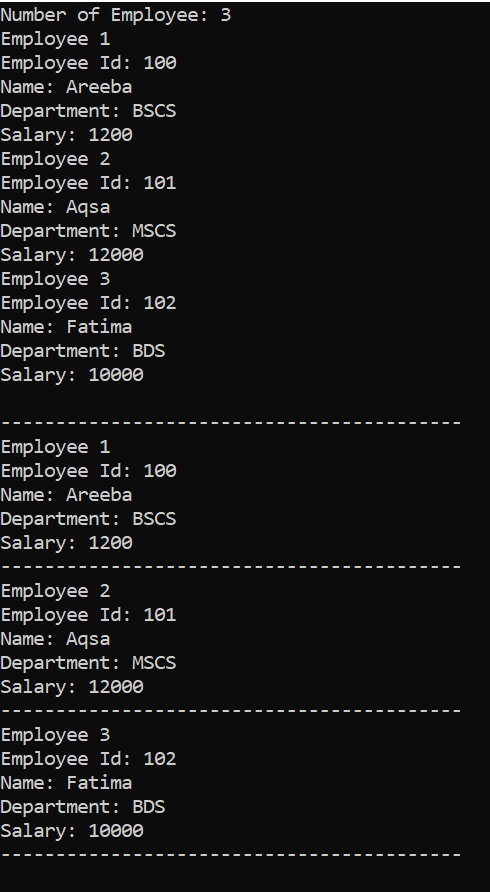
    }

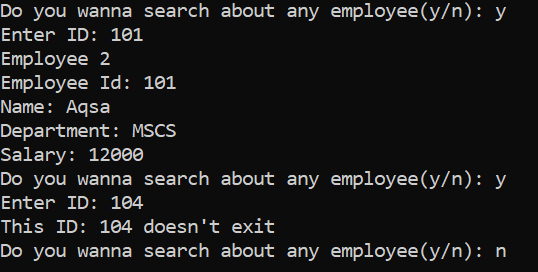
    }while(wanna != "n");

    delete[] emp;

}

**Output:**





**Task 04:**

struct Student{

   string name;

   int rollNum;

   int marks[5];

   float avg;

   int grades;

};

void Input(Student\* s, int num)

{   for(int i=0; i<num; i++)

    {   cout<<"Student "<<i+1<<endl;

        cout<<"Name: ";

        cin.ignore();

        getline(cin, s[i].name);

        cout<<"Roll Number: ";

        cin>>s[i].rollNum;

        for(int j=0; j<5; j++)

        {  cout<<"Maks of subject "<<j+1<<": ";

           cin>>s[i].marks[j];

        }

    }

    cout<<endl;

}

void Average\_Grades(Student\* s, int num)

{   int sum = 0;

    for(int i=0; i<num; i++)

    {  for(int j=0; j<5; j++)

       {  sum+=s[i].marks[j];

       }

       s[i].grades = sum;

       s[i].avg = float(sum) / 5;

       sum = 0;

    }

    for(int i=0; i<num; i++)

    {   cout<<"----------------------------"<<endl;

        cout<<"Student "<<i+1<<endl;

        cout<<"Average: "<<s[i].avg <<endl;

        cout<<"Grades: "<<s[i].grades<<endl;

        cout<<"----------------------------"<<endl;

    }

    cout<<endl;

}

int main()

{   int num;

    cout<<"Enter number of student: ";

    cin>>num;

    Student\* s = new Student[num];

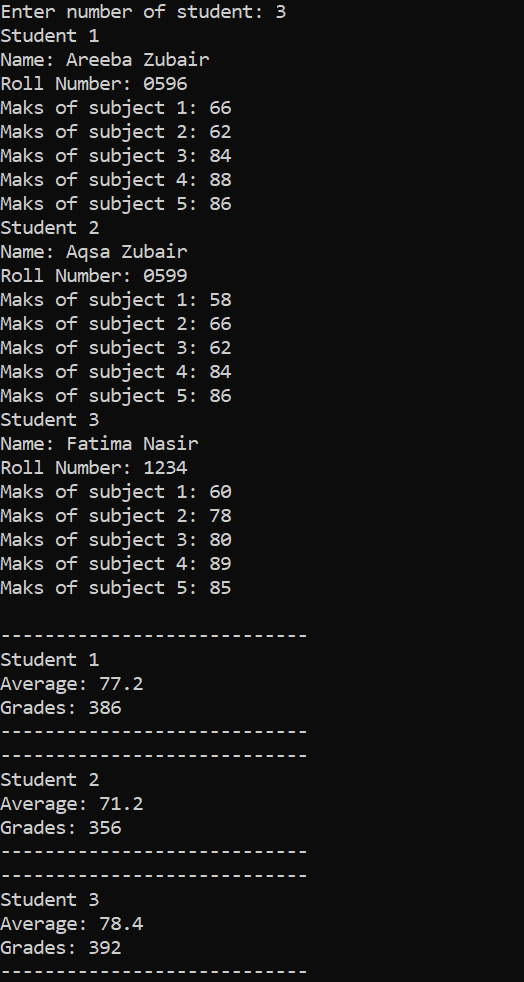
    Input(s,num);

    Average\_Grades(s,num);

    delete[] s;

}

**Output:**



**Task 05:**

void Comparision(char \*ptr1, char\* ptr2, int len)

{  int count = 0;

   for(int i=0; i<len; i++)

   {  if(ptr1[i] == ptr2[i])

      { count++;

      }

   }

   if(count == len)

   {   cout<<"Both string are equal\n";

   }else{

        cout<<"Both string are unequal\n";

   }

}

void Concatenation(char \*ptr1, char\* ptr2, int ptr1\_len, int length)

{  for(int i = ptr1\_len-1, j=0; i<length; i++)

   {  if(i==ptr1\_len-1)

      { ptr1[i] = ' ';

      }else{

        ptr1[i] = ptr2[j];

        j++;

      }

   }

   cout<<"Concatenated String: "<<ptr1<<endl;

}

int main()

{   char\* ptr1 = new char[100];

    char\* ptr2 = new char[100];

    cout<<"String 1: ";

    cin.getline(ptr1,100);

    cout<<"String 2: ";

    cin.getline(ptr2,100);

    cout<<"Length of String 1: "<< strlen(ptr1) <<endl;

    cout<<"Length of String 2: "<< strlen(ptr2) <<endl;

    if(strlen(ptr1) == strlen(ptr2))

    {  Comparision(ptr1,ptr2,strlen(ptr1));

    }

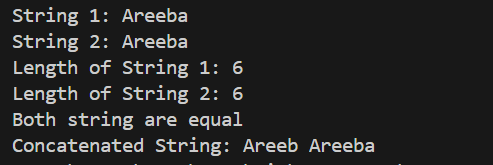
    Concatenation(ptr1,ptr2,strlen(ptr1),strlen(ptr1) + strlen(ptr2));

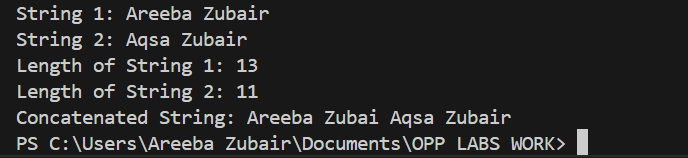
    delete[] ptr1;

    delete[] ptr2;

}

**Output 01:**



**Output 02:** 

**Task 06:**

struct Book{

    int bookId;

    string title;

    string author;

    bool isavailable;

};

void Add(Book\* book, int num)

{   for(int i=0; i<num; i++)

    {  cout<<"Book "<<i+1<<endl;

       cout<<"Id: ";

       cin>>book[i].bookId;

       cout<<"Title: ";

       cin.ignore();

       getline(cin,book[i].title);

       cout<<"Author: ";

       getline(cin,book[i].author);

       book[i].isavailable = 1;

    }

    cout<<endl;

}

void Remove(Book\* book, int &num)

{  int id, count = 0, found = 0;

   cout<<"Enter ID of the book to be removed: ";

   cin>>id;

   for(int i=0; i<num; i++)

   {  if(book[i].bookId == id)

      {   found++;

          continue;

      }else{

          book[count++] = book[i];

      }

   }

   if(!found){

       cout<<"Book with this Id: "<<id<<" is not available"<<endl;

   }else{

       num--;

       cout<<"Removed!\n";

   }

   cout<<endl;

}

void Display(Book\* book, int num)

{  for(int i=0; i<num; i++)

    {  cout<<"Book "<<i+1<<endl;

       cout<<"Id: "<<book[i].bookId<<endl;

       cout<<"Title: "<<book[i].title<<endl;

       cout<<"Author: "<<book[i].author<<endl;

       cout<<"------------------------------------\n";

    }

    cout<<endl;

}

void Search(Book\* book, int num)

{  int found =0;

   string title;

   cout<<"Enter Title of the book to be searched: ";

   cin.ignore();

   getline(cin,title);

   for(int i=0; i<num; i++)

   {  if(book[i].title == title)

      { found=1;

        cout<<"------------------------------------\n";

        cout<<"Id: "<<book[i].bookId<<endl;

        cout<<"Author: "<<book[i].author<<endl;

        cout<<"Available: Yes"<<endl;

        cout<<"------------------------------------\n";

      }

   }

    if(!found){

       cout<<"Book with this Title: "<<title<<" is not available"<<endl;

    }

   cout<<endl;

}

int main()

{  int num, removeNum;

   cout<<"Enter number of books to be added: ";

   cin>>num;

   Book\* book = new Book[num];

   Add(book,num);

   cout<<"Added List of Books:\n\n";

   Display(book,num);

   Remove(book,num);

   cout<<"Removed List of Books:\n\n";

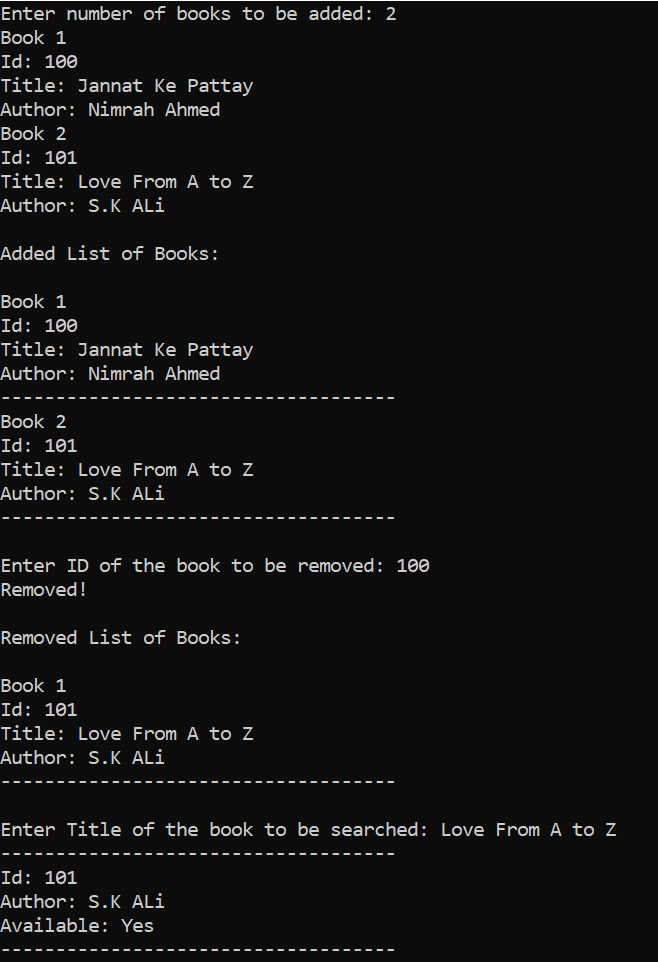
   Display(book,num);

   Search(book,num);

  delete[] book;

}

**Output:**



**Task 07:**

void Swap(int\* ptr1, int\* ptr2)

{  int\* temp = ptr1;

   ptr1 = ptr2;

   ptr2 = temp;

   cout<<"After Swapping\n";

   cout<<"Enter number 1: "<<\*ptr1<<endl;

   cout<<"Enter number 2: "<<\*ptr2<<endl<<endl;

}

void Reverse(int\* ptr, int num)

{  int temp;

   int j=num-1;

   for(int i=0; i<j; i++)

   {  temp = ptr[i];

      ptr[i] = ptr[j];

      ptr[j] = temp;

      j--;

   }

    cout<<"After Reversing\n";

    for(int i=0; i<num; i++)

    {  cout<<"Element "<<i+1<<": "<<ptr[i]<<endl;

    }

}

int main()

{   int num1, num2;

    cout<<"Enter number 1: ";

    cin>>num1;

    cout<<"Enter number 2: ";

    cin>>num2;

    int \*ptr1 = new int(num1);

    int \*ptr2 = new int(num2);

    Swap(ptr1,ptr2);

    cout<<"Enter the size of array: ";

    cin>>num1;

    int\* ptr = new int(num1);

    for(int i=0; i<num1; i++)

    {  cout<<"Element "<<i+1<<": ";

       cin>>\*(ptr + i);

    }

    Reverse(ptr,num1);

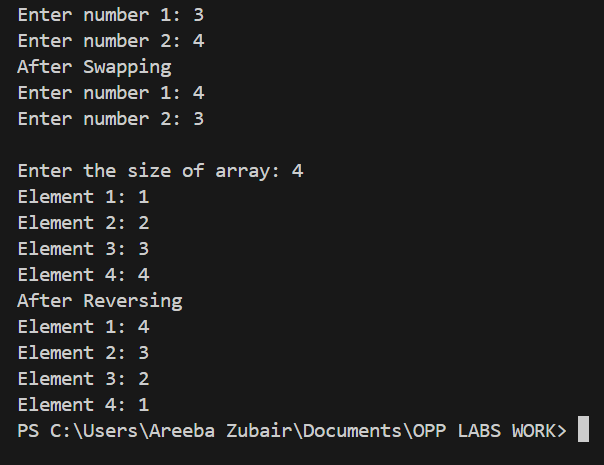
    delete[] ptr1;

    delete[] ptr2;

    delete[] ptr;

}

**Output:**



**Task 08:**

struct Product{

    int id;

    string name;

    int quantity;

    float price;

};

void Add(Product\* &p, int &num)

{   Product\* newP = new Product[num + 1];

    for (int i = 0; i < num; i++) {

        newP[i] = p[i];

    }

    cout<<"Id: ";

    cin>>newP[num].id;

    cout<<"Name: ";

    cin.ignore();

    getline(cin, newP[num].name);

    cout<<"Quantity: ";

    cin>>newP[num].quantity;

    cout<<"Price: ";

    cin>>newP[num].price;

    delete[] p;

    p = newP;

    num++;

    cout << "Product added successfully!\n";

    cout<<endl;

}

void Remove(Product\* &p, int& num, int id)

{   int found = -1;

    for(int i=0; i<num; i++)

    {  if(p[i].id == id)

        {  found = i;

           break;

        }

    }

    if(found == -1)

    {  cout<<"Product not found\n";

       return;

    }

    Product\* newP = new Product[num - 1];

    for(int i = 0, j = 0; i < num; i++) {

        if (i != found) {

            newP[j++] = p[i];

        }

    }

    delete[] p;

    p = newP;

    num--;

    cout << "Product removed successfully!\n\n";

}

void Update(Product\* &p, int &num, int id)

{   int found = 0 ;

    for(int i=0; i<num; i++)

    {  if(p[i].id == id)

       {    found = 1;

            cout<<"Name: ";

            cin.ignore();

            getline(cin, p[i].name);

            cout<<"Quantity: ";

            cin>>p[i].quantity;

            cout<<"Price: ";

            cin>>p[i].price;

       }

    }

    if(found == 0)

    {  cout<<"Product not found\n";

       return;

    }else{

        cout << "Product updated successfully!\n\n";

    }

}

void Display(Product\* p, int num)

{  cout<<"------------------------\n";

   for(int i=0; i<num; i++)

   {   cout<<"Product "<<i+1<<endl;

       cout<<"Id: "<<p[i].id<<endl;

       cout<<"Name: "<<p[i].name<<endl;

       cout<<"Quantity: "<<p[i].quantity<<endl;

       cout<<"Price: "<<p[i].price<<endl;

       cout<<"------------------------\n";

   }

   cout<<endl;

}

void TotalValue(Product\* p, int num)

{  int sum =0;

   cout<<"------------------------\n";

   for(int i=0; i<num; i++)

   {    cout<<"Product "<<i+1<<endl;

        cout<<"Total Value: "<<p[i].price\*p[i].quantity<<endl;

        cout<<"------------------------\n";

        sum+=p[i].price\*p[i].quantity;

   }

   cout<<"\nTotal Amount: "<<sum<<endl;

   cout<<endl;

}

int main()

{  int num = 0;

   Product\* p = new Product[num];

   int choise;

   int id;

   cout<<"Inventory Management\n";

   do

   {  cout<<"1-Add\n2-Remove\n3-Update\n4-Display\n5-Total Amount in System\n6-Exit\nEnter your choise: ";

      cin>>choise;

      switch(choise)

      {

        case 1:

        Add(p,num);

        break;

        case 2:

        cout<<"Id of the products to be removed: ";

        cin>>id;

        Remove(p,num,id);

        break;

        case 3:

        cout<<"Id of the products to be updated: ";

        cin>>id;

        Update(p,num,id);

        break;

        case 4:

        Display(p,num);

        break;

        case 5:

        TotalValue(p,num);

        break;

        case 6:

        cout<<"Thanks\n";

        break;

        default:

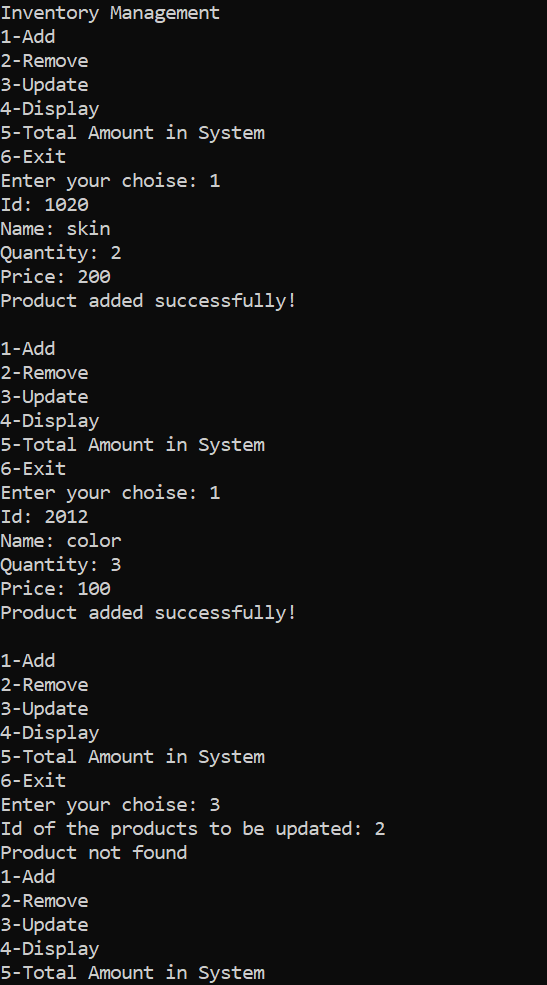
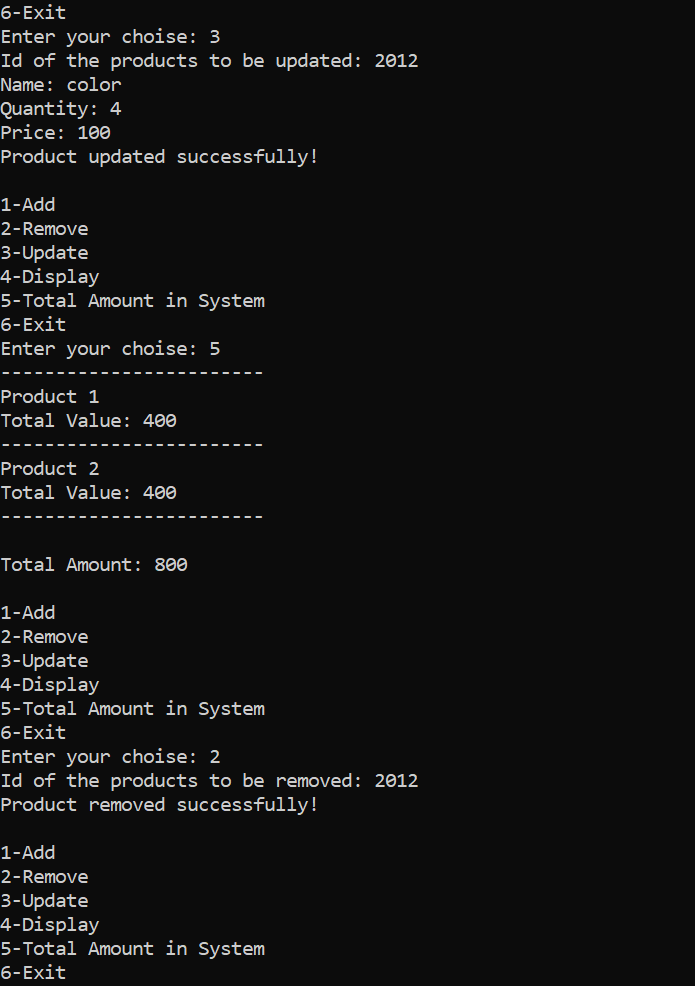
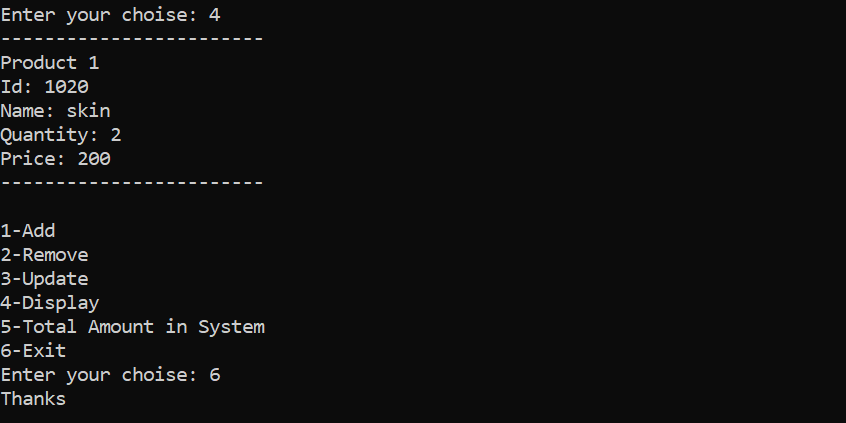
        cout<<"Error\n";

      }

   } while (choise != 6);

   delete[]  p;

}

**Output:**  

**Task 09:**

struct Student{

   string name;

   int rollNum;

   int marks;

};

void Input(Student\* s, int num)

{   for(int i=0; i<num; i++)

    {   cout<<"Student "<<i+1<<endl;

        cout<<"Name: ";

        cin.ignore();

        getline(cin, s[i].name);

        cout<<"Roll Number: ";

        cin>>s[i].rollNum;

        cout<<"Maks: ";

        cin>>s[i].marks;

    }

    cout<<endl;

}

void Highest(Student\* s, int num)

{  int max=0;

   string name;

   for(int i=0; i<num; i++)

   {  if(s[i].marks>max)

      {  max = s[i].marks;

         name = s[i].name;

      }

   }

   cout<<name<<" has the highest marks of "<<max<<endl;

   cout<<endl;

}

int main()

{   int num;

    cout<<"Enter number of student: ";

    cin>>num;

    Student\* s = new Student[num];

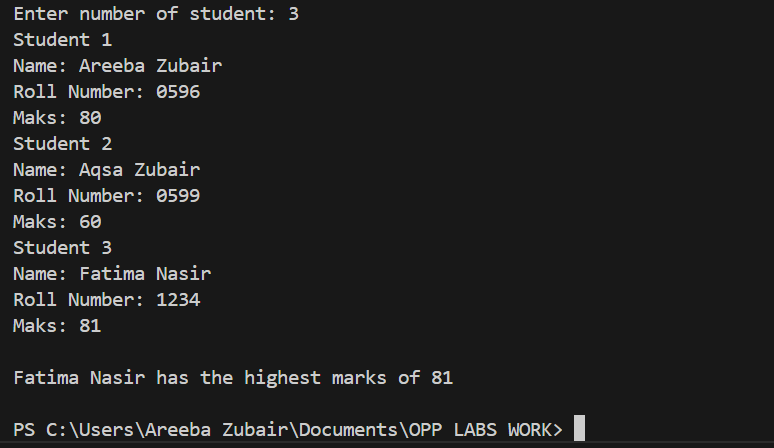
    Input(s,num);

    Highest(s,num);

    delete[] s;

}

**Output:**



**Task 10:**

void Input(int\*\* ptr, int r, int c)

{   for(int i=0; i<r; i++)

    {   for(int j=0; j<c; j++)

        {  cout<<"Element ["<<i<<"]["<<j<<"]: ";

           cin>>ptr[i][j];

        }

    }

}

int\*\* Transpose(int\*\* ptr, int r, int c)

{   int\*\* arr = new int\*[c];

    for(int i=0; i<c; i++)

    {  arr[i] = new int[r];

    }

    for(int i=0; i<2 ; i++)

    {   for(int j=0; j<3; j++)

        {  \*(\*(arr + j) + i) = \*(\*(ptr + i) + j);

        }

    }

    return arr;

}

void free(int\*\* ptr,int r)

{   for(int i=0; i<r; i++)

    {  delete[] ptr[i];

    }

    delete[] ptr;

}

int main()

{   int row, cols;

    cout<<"Enter rows: ";

    cin>>row;

    cout<<"Enter columns: ";

    cin>>cols;

    int\*\* ptr = new int\*[row];

    for(int i=0; i<row; i++)

    {  ptr[i] = new int[cols];

    }

    Input(ptr,row,cols);

    cout<<"Matrix"<<endl;

    for(int i=0; i<row ; i++)

    {   for(int j=0; j<cols; j++)

        {    cout<<\*(\*(ptr + i) + j)<<" ";

        }

        cout<<endl;

    }

    ptr =  Transpose(ptr,row,cols);

    cout<<"Transpose of matrix"<<endl;

    for(int i=0; i<cols ; i++)

    {   for(int j=0; j<row; j++)

        {    cout<<\*(\*(ptr + i) + j)<<" ";

        }

        cout<<endl;

    }

    free(ptr,row);

}

**Output:**

