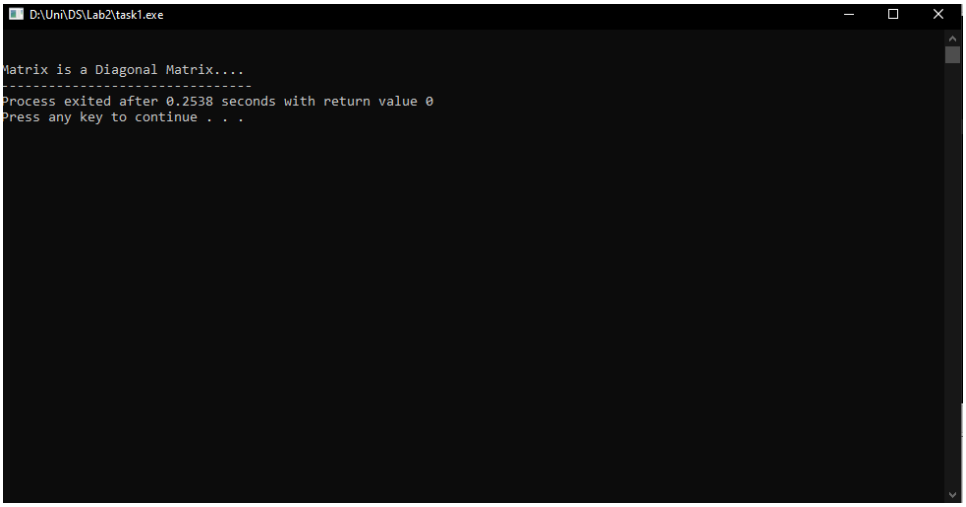
Lab 2

Syed Yousha

K226007

Task1:

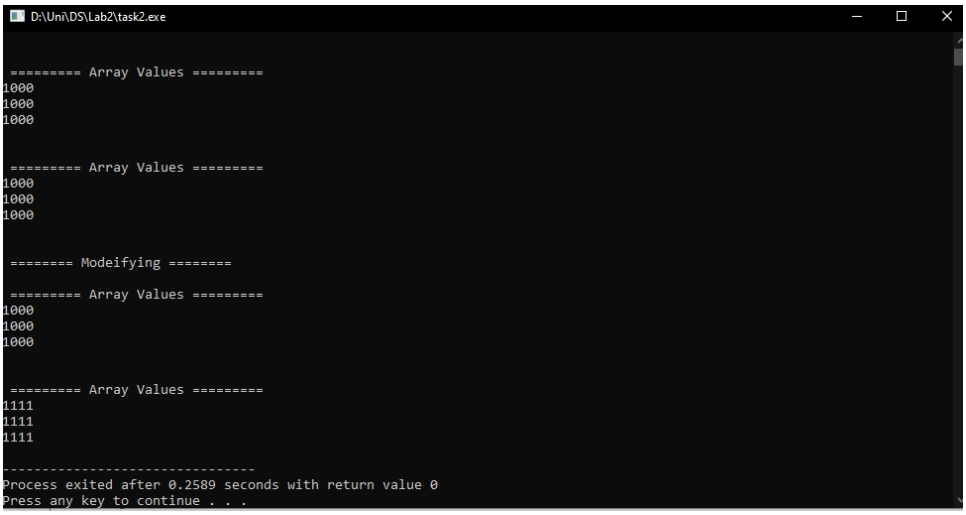



```
#include <iostream>

using namespace std;


int main()

{
```

```
int rows = 5, cols_size[5];
string id[5];

int* arr[rows];

for(int i=0; i<rows; i++)
{
cout<<"\nEnter your roll num: ";
cin>>id[i];

cout<<"\n Enter the number of courses: ";
cin>>cols_size[i];


arr[i] = new int[cols_size[i]];

//                  cout<<"\n\nEnter marks for "<<cols_size[i]<<" courses: ";

for(int j=0; j< cols_size[i]; j++)
{
cout<<"\n\n marks for course #"<<j+1<<" :" ;
cin>>arr[i][j];
}
}


cout<<"\n\n======== Students data ========";
```

```
for(int i=0; i<rows; i++)
{

cout<<"\n\n========== Student #"<<i+1<<" :" ;
cout<<"\nRoll num: "<<id[i];
cout<<"\nMarks:";
for(int j=0; j<cols_size[i]; j++)
{
cout<<arr[i][j]<<", ";

}
}

}
```

Task 2:




```
#include <iostream>
using namespace std;
```

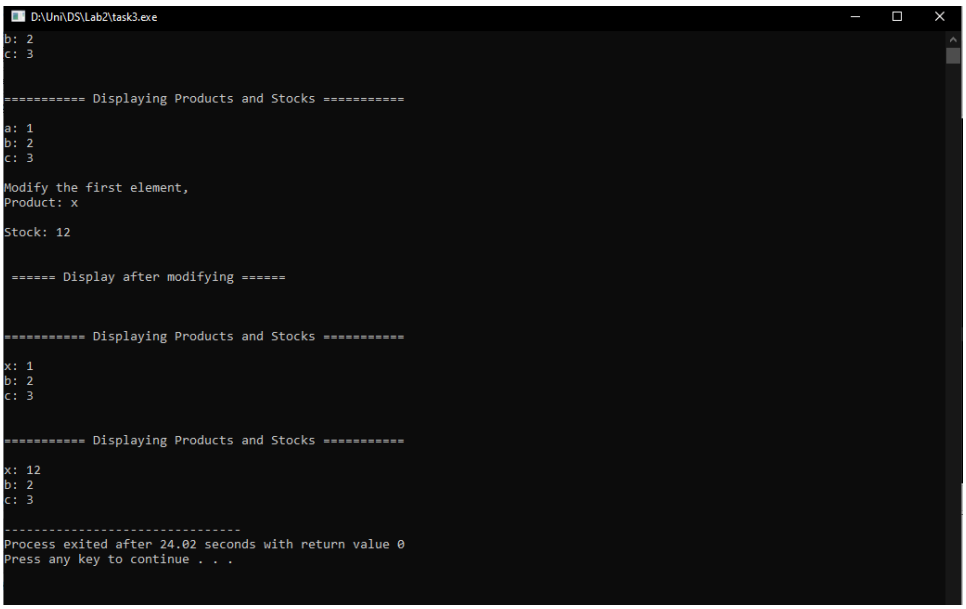
```
class StudentFeeManager
{
int *arr, size;
public:

StudentFeeManager(int s):size(s)
{}

void sett_array()
{
arr = new int[size];

for(int i=0; i<3; i++)
{
arr[i] = 1000;
}
}


void modify_array(int s)
{
arr = new int[s];
for(int i=0; i<3; i++)
{
arr[i] = 1111;
}
}
```

```
StudentFeeManager(StudentFeeManager &obj)
{
arr = new int[size];
memcpy(arr, obj.arr, 3*sizeof(int));
}

StudentFeeManager& operator = (const StudentFeeManager &obj)
{
arr = new int[size];
if( obj.arr != arr)
{
memcpy(arr, obj.arr, 3*sizeof(int));
}

return *this;
}



int &operator [](int i)
{
if(i<0 || i>2)
{
cout<<"\nError! array out of bound!";
exit(1);
}
}
```

```
void display()
{

cout<<"\n\n ========= Array Values =========\n";
for(int i=0; i<3; i++)
{
cout << arr[i]<<endl;
}
}

~StudentFeeManager()
{
delete[] arr;
}

};

int main()
{

StudentFeeManager obj(3);

obj.sett_array();
obj.display();

StudentFeeManager obj2(obj);
```

```
obj2.display();



cout<<"\n\n ======== Modeifying ========";
obj2.modify_array(3);
obj.display();
obj2.display();
}
```

Task 3:




```
#include <iostream>
#include <cstring>
using namespace std;

class ProductStockManger
{
```

```
    string *product;
    int *stock, size;

public:
    ProductStockManger() {}
    ProductStockManger(int s) : size(s)
    {
        product = new string[size];
        stock = new int[size];

        for (int i = 0; i < size; i++)
        {
            cout << "\nEnter the name of product #" << i + 1 << " : ";
            cin >> product[i];

            cout << "\nEnter the number of stock of product #" << i + 1 << " : ";
            cin >> stock[i];
        }
    }

    // Copy constructore
    ProductStockManger(const ProductStockManger &obj) : size(obj.size)
    {
        product = new string[size];
        stock = new int[size];

        memcpy(product, obj.product, size * sizeof(string));
```

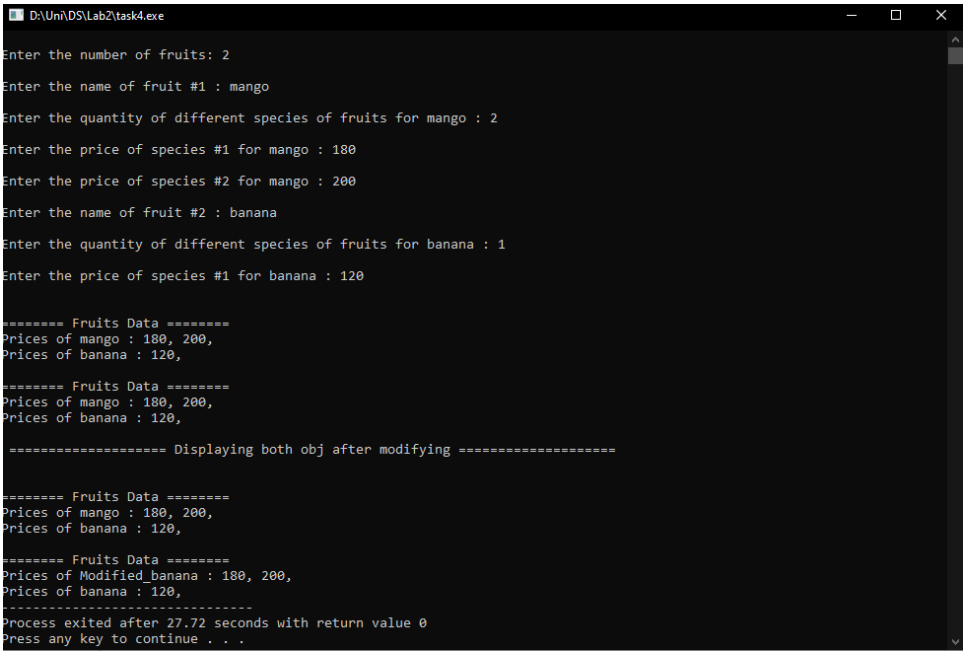
```
    memcpy(stock, obj.stock, size * sizeof(int));
}

// k226007 Copy assignment operator
ProductStockManger &operator=(const ProductStockManger &obj)
{
    if (this == &obj)
    {
        return *this;
    }

    delete[] product;
    delete[] stock;
    product = NULL;
    stock = NULL;

    size = obj.size;
    product = new string[size];
    stock = new int[size];

    memcpy(product, obj.product, size * sizeof(string));
    memcpy(stock, obj.stock, size * sizeof(int));

    return *this;
}

// Safe array
```

```
void operator[](int i)
{
   if (i < 0 || i > size)
   {
      cout << "\n Boundary Error!!";
      exit(1);
   }
}

// Methods
void modify()
{
   cout << "\nModify the first element,";
   cout << "\nProduct: ";
   cin >> product[0];
   cout << "\nStock: ";
   cin >> stock[0];
}

void display()
{

   cout << "\n\n========== Displaying Products and Stocks ==========\n\n";
   for (int i = 0; i < size; i++)
   {
      cout << product[i] << ": " << stock[i] << endl;
   }
```

```
    }

    ~ProductStockManger()
    {
        delete[] product;
        product = NULL;
        delete[] stock;
        stock = NULL;
    }
};

int main()
{
    ProductStockManger obj(3);
    obj.display();

    ProductStockManger obj2(obj);
    obj2.display();

    // Modifying
    obj2.modify();

    cout << "\n\n ====== Display after modifying ======\n\n";
    obj.display();
    obj2.display();

    return 0;
```

}

Task4:




```
#include <iostream>
#include <cstring>
using namespace std;

class FruitInventory
{
    int *species;
    int fruits;
    int **arr;
    string *fruit_names;

public:
    FruitInventory() {}
```

```
FruitInventory(int q) : fruits(q)
{
   arr = new int *[fruits];
   fruit_names = new string[fruits];
   species = new int[fruits];

   for (int i = 0; i < fruits; i++)
   {
      cout << "\nEnter the name of fruit #" << i + 1 << " : ";
      cin >> fruit_names[i];

      cout << "\nEnter the quantity of different species of fruits for " << fruit_names[i] << " : ";
      cin >> species[i];

      arr[i] = new int[species[i]];
      for (int j = 0; j < species[i]; j++)
      {
         cout << "\nEnter the price of species #" << j + 1 << " for " << fruit_names[i] << " : ";
         cin >> arr[i][j];
      }
   }
}

int &operator()(int i)
{
   if (i < 0 || i >= fruits)
   {
```

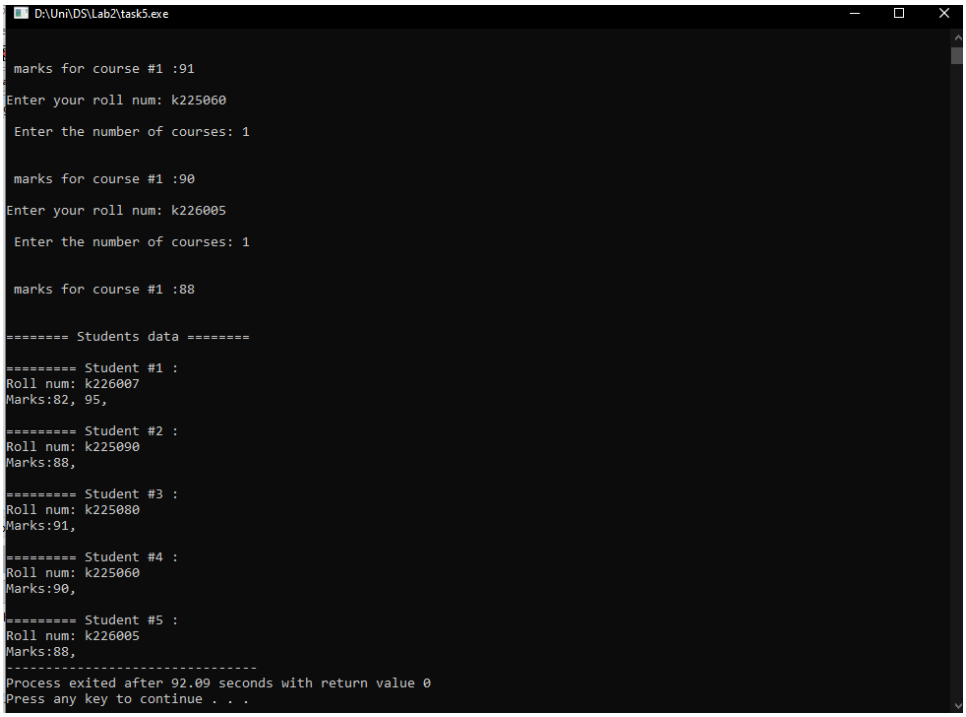
```
        cout << "\nBoundary error!";
        exit(1);
    }
    return species[i];
}

void displayInvent()
{
    cout << "\n\n======== Fruits Data ========";
    for (int i = 0; i < fruits; i++)
    {
        cout << "\nPrices of " << fruit_names[i] << " : ";

        for (int j = 0; j < species[i]; j++)
        {
            cout << arr[i][j] << ", ";
        }
    }
}

void modify()
{
    fruit_names[0] = "Modified_banana";
}

FruitInventory(const FruitInventory &obj) : fruits(obj.fruits), fruit_names(new string[obj.fruits])
{
```

```
    species = new int[fruits];
    arr = new int *[fruits];

    for (int i = 0; i < fruits; i++)
    {
        fruit_names[i] = obj.fruit_names[i];
        species[i] = obj.species[i];

        arr[i] = new int[species[i]];
        memcpy(arr[i], obj.arr[i], sizeof(int) * species[i]);
    }
}

FruitInventory &operator=(const FruitInventory &obj)
{
    if (this == &obj)
        return *this;

    for (int i = 0; i < fruits; i++)
    {
        delete[] arr[i];
    }
    delete[] arr;
    delete[] species;
    delete[] fruit_names;

    fruits = obj.fruits;
```

```
        fruit_names = new string[fruits];
        species = new int[fruits];
        arr = new int *[fruits];

        for (int i = 0; i < fruits; i++)
        {
            species[i] = obj.species[i];
            fruit_names[i] = obj.fruit_names[i];

            arr[i] = new int[species[i]];
            memcpy(arr[i], obj.arr[i], sizeof(int) * species[i]);
        }

        return *this;
    }

    ~FruitInventory()
    {
        for (int i = 0; i < fruits; i++)
        {
            delete[] arr[i];
        }
        delete[] arr;
        delete[] species;
        delete[] fruit_names;
    }
};
```

```
int main()
{
   int fruits;
   cout << "\nEnter the number of fruits: ";
   cin >> fruits;

   FruitInventory obj(fruits);
   obj.displayInvent();

   // Object 2
   FruitInventory obj2(obj);
   obj2.displayInvent();

   // Modifying (k226007)
   obj2.modify();
   cout << "\n\n ==================== Displaying both obj after modifying ====================\n";
   obj.displayInvent();
   obj2.displayInvent();
   return 0;
}
```

Task 5:




```
#include <iostream>
using namespace std;

int main()
{
int rows = 5, cols_size[5];
string id[5];

int* arr[rows];

for(int i=0; i<rows; i++)
{
cout<<"\nEnter your roll num: ";
cin>>id[i];
```

```
cout<<"\n Enter the number of courses: ";
cin>>cols_size[i];


arr[i] = new int[cols_size[i]];

//                  cout<<"\n\nEnter marks for "<<cols_size[i]<<" courses: ";

for(int j=0; j< cols_size[i]; j++)
{
cout<<"\n\n marks for course #"<<j+1<<" :" ;
cin>>arr[i][j];
}
}


cout<<"\n\n======== Students data ========";
for(int i=0; i<rows; i++)
{

cout<<"\n\n========= Student #"<<i+1<<" :" ;
cout<<"\nRoll num: "<<id[i];
cout<<"\nMarks:";
for(int j=0; j<cols_size[i]; j++)
{
cout<<arr[i][j]<<", ";
```

```
}
}


}
```