„Winery Production”

Made by: Muhammad Eman Aftab

Neptun code: IJE4R1

E-mail: *emanaftab2022@gmail.com*

Course code: 2

Teacher’s name: Vincze Dorottya

10/1/2024

Content

User documentation

Task

Multiple wineries register the type and amount of wine sold, and the price the wine was sold at each year. Wineries and products can be listed multiple times.

Write a program that gives the name of the winery, the product, and the amount sold for the highest price by any of the wineries

Usage

Runtime environment

Hp PC, an operating system capable of running exe files (eg. Windows 7,10). Visual studio (2022) developer tool.

Starting the program

The program can be found in the archived file by the name `WineryProduction\Program\bin\Release\net6.0\B_3.exe`. You can start the program by clicking the `B_3.exe` file.

Program input

The program reads the input data from the keyboard in the following order:

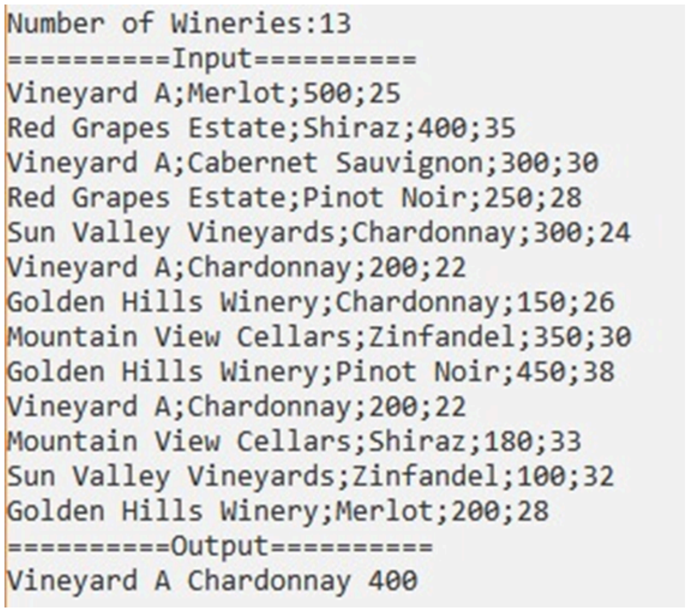
#	Data	Explanation
1.	*N*	The count of years ($1 \leq N \leq 100$).
2.	*$winery_1$name, $winery_1$product, $winery_1$amount, $winery_1$ price*	The amount of $winery_1$ ($1 \leq amount_1 \leq 10\ 000$). The price of $winery_1$ ($1 \leq price_1 \leq 10\ 000$).
3.	*$winery_2$name, $winery_2$product, $winery_2$amount $winery_2$ price*	The amount of $winery_2$ ($1 \leq amount_2 \leq 10\ 000$). The price of $winery_2$ ($1 \leq price_2 \leq 10\ 000$)
...	...	
N	*$winery_N$.name, $winery_N$.product, $winery_N$.amount, $winery_N$.price*	The amount of Nth $winery_1$ ($1 \leq amount_N \leq 10\ 000$). The price of Nth $winery_1$ ($1 \leq price_N \leq 10\ 000$)

N+1	*winery$_{N+1}$.name, winery$_{N+1}$.product, winery$_{N+1}$.amount, winery$_{N+1}$.price*	The amount of N+1 winery$_1$ ($1 \leq \text{amount}_{N+1} \leq 10\,000$). The price of N+1 winery$_1$ ($1 \leq \text{price}_{N+1} \leq 10\,000$)

Program output

The first line of the standard output should contain the name of the winery, the product, and the amount sold for the highest price by any of the wineries. If there are multiple products sold for the same highest price give the first occurrence.

Sample input and output

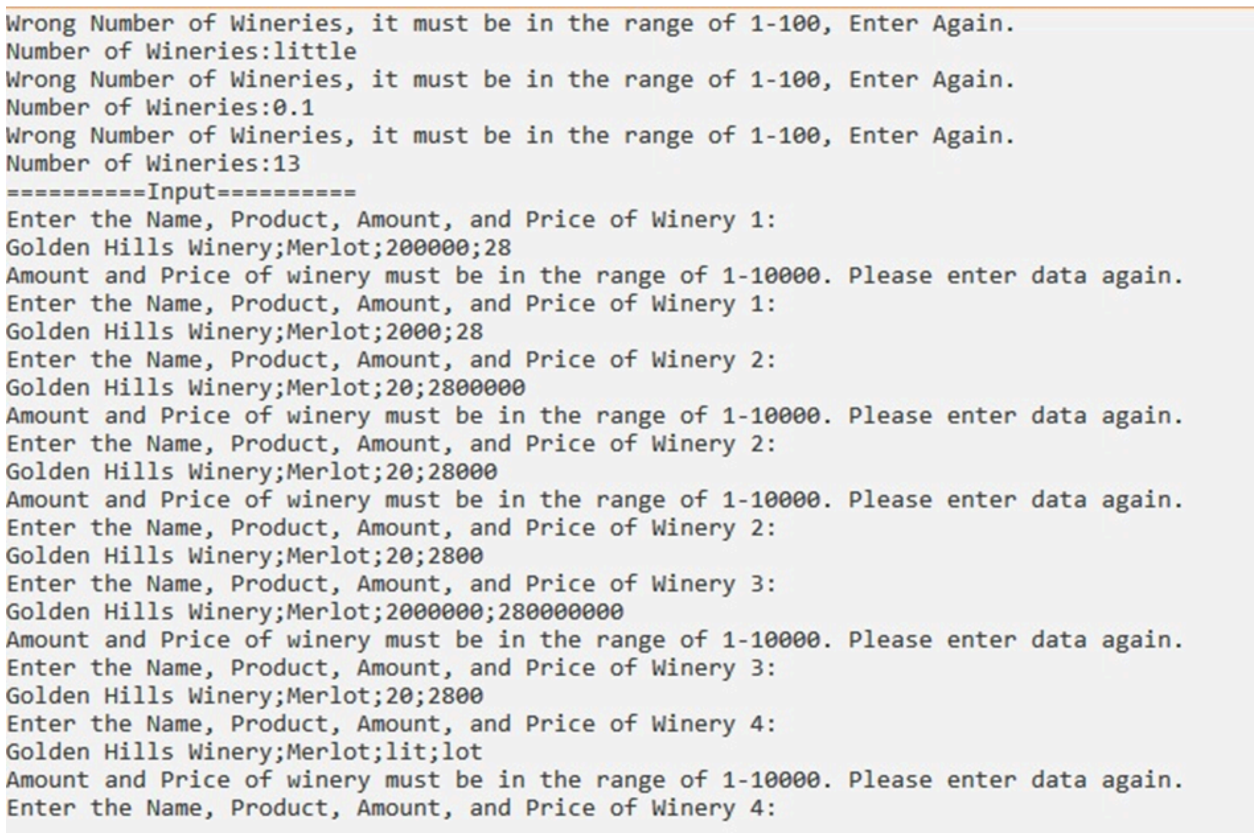


```
Number of Wineries:13
==========Input==========
Vineyard A;Merlot;500;25
Red Grapes Estate;Shiraz;400;35
Vineyard A;Cabernet Sauvignon;300;30
Red Grapes Estate;Pinot Noir;250;28
Sun Valley Vineyards;Chardonnay;300;24
Vineyard A;Chardonnay;200;22
Golden Hills Winery;Chardonnay;150;26
Mountain View Cellars;Zinfandel;350;30
Golden Hills Winery;Pinot Noir;450;38
Vineyard A;Chardonnay;200;22
Mountain View Cellars;Shiraz;180;33
Sun Valley Vineyards;Zinfandel;100;32
Golden Hills Winery;Merlot;200;28
==========Output==========
Vineyard A Chardonnay 400
```

Possible errors

The input should be given according to the sample. If the number of wineries is not a whole number, or it is not in the range 1..100, it will cause a problem. If one of the amount or price of winery is not a number, or it is not in the range 1..10,000, it will also cause a problem. In the case of an error, the program displays an error message, or asks for the repetition of the input.

Sample of running in the case of invalid data:



```
Wrong Number of Wineries, it must be in the range of 1-100, Enter Again.
Number of Wineries:little
Wrong Number of Wineries, it must be in the range of 1-100, Enter Again.
Number of Wineries:0.1
Wrong Number of Wineries, it must be in the range of 1-100, Enter Again.
Number of Wineries:13
==========Input==========
Enter the Name, Product, Amount, and Price of Winery 1:
Golden Hills Winery;Merlot;200000;28
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 1:
Golden Hills Winery;Merlot;2000;28
Enter the Name, Product, Amount, and Price of Winery 2:
Golden Hills Winery;Merlot;20;2800000
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 2:
Golden Hills Winery;Merlot;20;28000
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 2:
Golden Hills Winery;Merlot;20;2800
Enter the Name, Product, Amount, and Price of Winery 3:
Golden Hills Winery;Merlot;2000000;280000000
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 3:
Golden Hills Winery;Merlot;20;2800
Enter the Name, Product, Amount, and Price of Winery 4:
Golden Hills Winery;Merlot;lit;lot
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 4:
```

Developer documentation

Task

Multiple wineries register the type and amount of wine sold, and the price the wine was sold at each year. Wineries and products can be listed multiple times.
Write a program that gives the name of the winery, the product, and the amount sold for the highest price by any of the wineries.

Specification

Input: $N \in \mathbb{N}$, wines $[1..N] \in$ **wineN**
wine = (name $\times$ type $\times$ amount $\times$ price),
name = T, type = T, amount = N, price = N

Output: result $\in$ **outputN**,
output = (name $\times$ type $\times$ amount),
name = T, type = T, amount = N

Precondition: $1 \le N \le 100$
and $\forall i\ (1 \le i \le N) : 1 \le wines_i.amount \le 10{,}000$
and $\forall i\ (1 \le i \le N) : 1 \le wines_i.price \le 10{,}000$
Postcondition:

$$wines[\,] = \underset{i=1}{\overset{cnt}{\exists}} (wines_i.name = name \text{ And } wines_i.type = type)$$

$$wines[\,] = \sum_{i=1}^{cnt} (wines_i.amount + amount \text{ And } wines_i.price + price)$$

$$result = \underset{i=1}{\overset{cnt}{MAX}} (wines_i.price)$$

Comment: If there are less than 2 wineries, the program will write out that winery, and not the logical value (as it was required by the task).

Developer environment

Hp PC, an operating system capable of running exe files (eg. Windows 7,10). Visual studio (2022) developer tool.

Source code

All the sources can be found in the `Winery Production` folder (after extraction). The folder structure used for development:

File	Explanation
WineryProduct\Program\bin\Release\net6.0\B_3.exe	Executable code
WineryProduct\Program\obj\Release\net6.0\B_3.pdb,B_3.dll	Semi-compiled code
WineryProduct\Program\Program.cs	C# source code
WineryProduct\Program\B_3.sln	Visual Studio Solution
WineryProduct\Program\TestCases\inp1.txt	input test file$_1$
WineryProduct\Program\TestCases\inp2.txt	input test file$_2$
WineryProduct\Program\TestCases\outp1.txt	output test file$_1$
WineryProduct\Program\TestCases\outp2.txt	output test file$_2$
WineryProduct\Documentation\WineryProduction.docx	documentation (this file)

Solution

Program parameters

Types

```
wine   = Record (name,type : String , amount, price : Integer)
output = Record (name,type : String , amount : Integer)
```

Variables

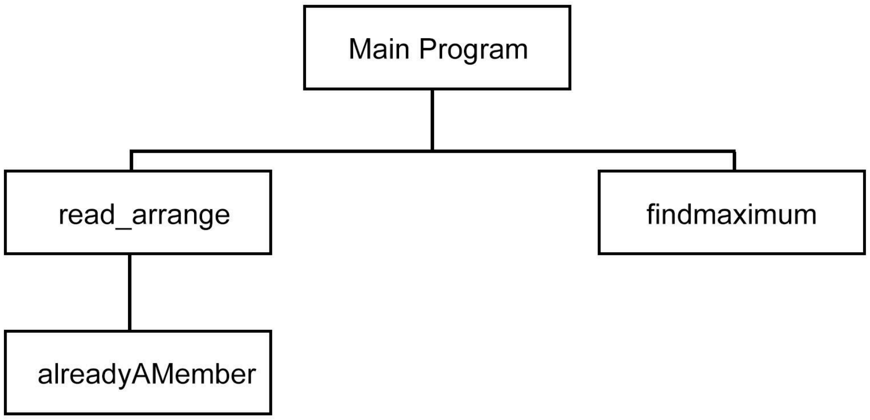
```
N          : Integer
wines      : wine
result     : output
```

The structure of the program

The modules used by the program, and their locations:

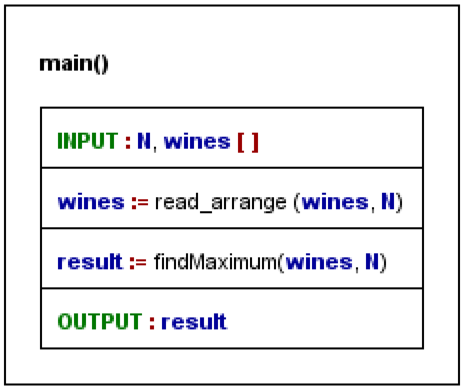
`Program.cs`	– the program file, in the Program folder
`B_3.csproj`	–C# Project File
`Program.sln`	– Visual Studio File

Structure of functions



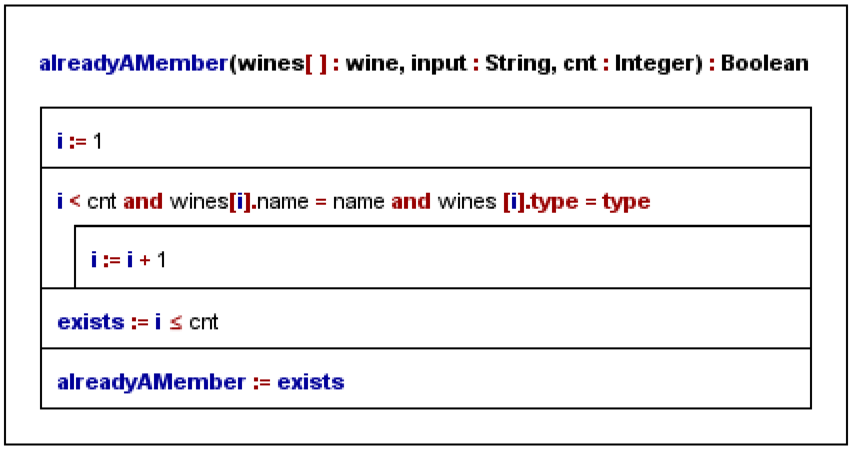
The algorithm of the program

Main program:



```
main()
INPUT : N, wines [ ]
wines := read_arrange (wines, N)
result := findMaximum(wines, N)
OUTPUT : result
```

Subprograms:



```
alreadyAMember(wines[ ] : wine, input : String, cnt : Integer) : Boolean
i := 1
i < cnt and wines[i].name = name and wines [i].type = type
    i := i + 1
exists := i ≤ cnt
alreadyAMember := exists
```

```
read_arrange(wines[ ] : wine, N : Integer) : wine [ ]

cnt := 0
for i ← 0 to N
    input = Data From User
    alreadyAMember(input, wines, cnt)
    T:
        for j ← 0 to cnt
            wines[j].name = input[0] AND wines[j].type = input[1]
            T:
                wines[j].amount += input[2]
                wines[j].price += input[3]
            F:
                ∅
    F:
        wines[cnt].name = input[0]
        wines[cnt].type = input[1]
        wines[cnt].amount = input[2]
        wines[cnt].price = input[3]
        cnt++
read_arrange := wines
```

```
findMaximum(wines[ ] : wine, cnt : Integer) : Output

max_result.name = wines[0].name
max_result.amount = wines[0].amount
max_result.type = wines[0].type;
maxPrice = wines[0].price
for i ← 1 to cnt
    wines[i].price > maxPrice
    T:
        max_result.name = wines[i].name
        max_result.amount = wines[i].amount
        max_result.type = wines[i].type
        maxPrice = wines[i].price
    F:
        ∅
findMaximum := max_result
```

The code

The content of the `program.cs` file:

```
/*

  Created by: Muhammad Eman Aftab

  Neptun: IJE4R1

  E-mail: emanaftab2022@gmail.com

  Task: „Winery Production”

*/

using System;

namespace B_3

{

    internal class Program

    {

        public struct wine

        {

            public string name;

            public string type;

            public int amount;

            public int price;

        }


        public struct Output

        {

            public string name;

            public string type;

            public int amount;
        }


        static void Main(string[] args)

        {

            bool good;

            int N;

            do

            {

                Console.Write("Number of Wineries:");

                string n_winery = Console.ReadLine();

                good = int.TryParse(n_winery, out N) && N >= 1 && N <= 100;

                if (!good)

                {



                    Console.WriteLine("Wrong Number of Wineries, it must be in
the range of 1-100, Enter Again.");

                }

            }



            while (!(good));



            wine[] wines = new wine[N];

            wines = read_arrange(wines, N);

            Output result = findMaximum(wines, N);

            Console.WriteLine("==========Output==========");

            Console.WriteLine($"{result.name} {result.type} {result.amount}");
            //string output = result.name + " " + result.type + " " +
result.amount;

            Console.ReadLine();




        }


        public static bool alreadyAMember(string input, wine[] wines, int cnt)

        {

            int i = 0;

            string name = input.Split(";")[0];

            string type = input.Split(";")[1];

            while (i < cnt && !(wines[i].name == name && wines[i].type == type)
)

            {

                i = i + 1;

            }

            bool exists = i < cnt;

            return exists;

        }


        public static Output findMaximum(wine[] wines, int cnt)

        {

            Output max_result;

            max_result.name = wines[0].name;

            max_result.amount = wines[0].amount;

            max_result.type = wines[0].type;
```

```
    int maxPrice = wines[0].price;

    for (int i = 0; i < cnt; i++)
    {
        if (wines[i].price > maxPrice)
        {
            max_result.name = wines[i].name;
            max_result.amount = wines[i].amount;
            max_result.type = wines[i].type;
            maxPrice = wines[i].price;
        }
    }
    return max_result;
}

public static wine[] read_arrange(wine[] wines, int N)
{
    Console.WriteLine("==========Input==========");
    int cnt = 0;

    for (int i = 0; i < N; i++)
    {
        string input;
        bool good;
        do
        {
```

```
                    Console.WriteLine("Enter the Name, Product, Amount, and
Price of Winery {0}:",i+1);

                    input = Console.ReadLine();

                    string[] inputValues = input.Split(';');



                    int amount, price = 0;



                    good = int.TryParse(inputValues[2], out amount) && amount
>= 1 && amount <= 10000 &&

                           int.TryParse(inputValues[3], out price) && price >=
1 && price <= 10000;



                    if (!good)

                    {

                        Console.WriteLine("Amount and Price of winery must be
in the range of 1-10000. Please enter data again.");

                    }

                    else

                    {

                        wines[i].amount = amount;

                        wines[i].price = price;

                    }

                }

                while (!good);
                if (alreadyAMember(input, wines, cnt))

                {

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

                    {

                        if (wines[j].name == input.Split(";")[0] &&
wines[j].type == input.Split(";")[1])

                        {

                            int amount = Convert.ToInt32(input.Split(";")[2]);

                            int price = Convert.ToInt32(input.Split(";")[3]);

                            wines[j].amount += amount;

                            wines[j].price += price;

                        }

                    }

                }

                else

                {

                    wines[cnt].name = input.Split(";")[0];

                    wines[cnt].type = input.Split(";")[1];

                    wines[cnt].amount = Convert.ToInt32(input.Split(";")[2]);

                    wines[cnt].price = Convert.ToInt32(input.Split(";")[3]);

                    cnt++;

                }

            }

            return wines;

        }

    }
```

```
}
```

Testing

Valid test cases

1. test case: in1.txt

Input

```
Number of Wineries:13

Enter the Name, Product, Amount, and Price of Winery 1:
Vineyard A;Merlot;500;25
Enter the Name, Product, Amount, and Price of Winery 2:
Red Grapes Estate;Shiraz;400;35
Enter the Name, Product, Amount, and Price of Winery 3:
Vineyard A;Cabernet Sauvignon;300;30
Enter the Name, Product, Amount, and Price of Winery 4:
Red Grapes Estate;Pinot Noir;250;28
Enter the Name, Product, Amount, and Price of Winery 5:
Sun Valley Vineyards;Chardonnay;300;24
Enter the Name, Product, Amount, and Price of Winery 6:
Vineyard A;Chardonnay;200;22
Enter the Name, Product, Amount, and Price of Winery 7:
Golden Hills Winery;Chardonnay;150;26
Enter the Name, Product, Amount, and Price of Winery 8:
Mountain View Cellars;Zinfandel;350;30
Enter the Name, Product, Amount, and Price of Winery 9:
Golden Hills Winery;Pinot Noir;450;38
Enter the Name, Product, Amount, and Price of Winery 10:
Vineyard A;Chardonnay;200;22
Enter the Name, Product, Amount, and Price of Winery 11:
Mountain View Cellars;Shiraz;180;33
Enter the Name, Product, Amount, and Price of Winery 12:
Sun Valley Vineyards;Zinfandel;100;32
Enter the Name, Product, Amount, and Price of Winery 13:
Golden Hills Winery;Merlot;200;28
```

Output

```
Vineyard A Chardonnay 400
```

2. *test case: in2.txt*

Input – *starts with continent, there are at least 2 islands*

```
Number of Wineries:15

Enter the Name, Product, Amount, and Price of Winery 1:
Vineyard A;Merlot;500;25
Enter the Name, Product, Amount, and Price of Winery 2:
Red Grapes Estate;Shiraz;400;35
Enter the Name, Product, Amount, and Price of Winery 3:
Vineyard A;Cabernet Sauvignon;300;30
Enter the Name, Product, Amount, and Price of Winery 4:
Red Grapes Estate;Pinot Noir;250;28
Enter the Name, Product, Amount, and Price of Winery 5:
Sun Valley Vineyards;Chardonnay;300;24
Enter the Name, Product, Amount, and Price of Winery 6:
Golden Hills Winery;Chardonnay;150;26
Enter the Name, Product, Amount, and Price of Winery 7:
Mountain View Cellars;Zinfandel;350;30
Enter the Name, Product, Amount, and Price of Winery 8:
Sun Valley Vineyards;Sauvignon Blanc;200;18
Enter the Name, Product, Amount, and Price of Winery 9:
Golden Hills Winery;Pinot Noir;450;38
Enter the Name, Product, Amount, and Price of Winery 10:
Vineyard A;Chardonnay;200;22
Enter the Name, Product, Amount, and Price of Winery 11:
Mountain View Cellars;Shiraz;180;33
Enter the Name, Product, Amount, and Price of Winery 12:
Red Grapes Estate;Cabernet Sauvignon;150;40
Enter the Name, Product, Amount, and Price of Winery 13:
Sun Valley Vineyards;Zinfandel;100;32
Enter the Name, Product, Amount, and Price of Winery 14:
Golden Hills Winery;Merlot;200;28
Enter the Name, Product, Amount, and Price of Winery 15:
Mountain View Cellars;Sauvignon Blanc;120;20
```

Output
Red Grapes Estate Cabernet Sauvignon 150

Invalid test cases

3. test case

Input – *wrong number of wineries*
Number of Wineries:little
Output
Asking again: N =

4. test case

Intput – wrong *height*
Number of Wineries:13 Enter the Name, Product, Amount, and Price of Winery 1: Vineyard A;Merlot;little;25
Output
Asking again: Amount and Price of winery must be in the range of 1-10000. Please enter data again. Enter the Name, Product, Amount, and Price of Winery 1:

5. test case

Input – *wrong number of wineries*
Number of Wineries:13 Enter the Name, Product, Amount, and Price of Winery 1: Vineyard A;Merlot;500;250000
Output

```
Asking again:
Amount and Price of winery must be in the range of 1-10000. Please enter data again.
Enter the Name, Product, Amount, and Price of Winery 1
```

…

6. test case

…

Further development options

1. Data to be read from file
2. Detection of wrong file input, writing out the location and ID# of error
3. Capability to run multiple times after each other
4. Visual representation of input data, and emphasizing the result wineries with different colors.