

Kauno technologijos universitetas

Informatikos fakultetas

Objektinis programavimas I (P175B118)

Laboratorinių darbų ataskaita

Nojus Raškevičius, IFF-0/6.

Lekt. Vacius Jusas

Kaunas 2020

TURINYS

1. Duomenų klasė 3

1.1. Darbo užduotis 3

1.2. Programos tekstas 4

1.3. Pradiniai duomenys ir rezultatai 10

1.4. Dėstytojo pastabos 12

2. Skaičiavimų klasė 13

2.1. Darbo užduotis 13

2.2. Programos tekstas 13

2.3. Pradiniai duomenys ir rezultatai 13

2.4. Dėstytojo pastabos 13

3. Konteineris 14

3.1. Darbo užduotis 14

3.2. Programos tekstas 14

3.3. Pradiniai duomenys ir rezultatai 14

3.4. Dėstytojo pastabos 14

4. Teksto analizė ir redagavimas 15

4.1. Darbo užduotis 15

4.2. Programos tekstas 15

4.3. Pradiniai duomenys ir rezultatai 15

4.4. Dėstytojo pastabos 15

5. Paveldėjimas 16

5.1. Darbo užduotis 16

5.2. Programos tekstas 16

5.3. Pradiniai duomenys ir rezultatai 16

5.4. Dėstytojo pastabos 16

# Duomenų klasė

## Darbo užduotis

**Kompiuterinis žaidimas**. Kuriate „fantasy“ tipo kompiuterinį žaidimą. Duomenų faile turite informacija apie žaidimo herojus: vardas, rasė, klasė, gyvybės taškai, mana, žalos taškai, gynybos taškai, jėga, vikrumas, intelektas, ypatinga galia.

* Raskite daugiausiai gyvybės taškų turintį herojų, ekrane atspausdinkite jo vardą, rasę, klasę ir gyvybės taškų kiekį. Jei yra keli, spausdinkite visus.
* Raskite žaidėją, kurio gynybos ir žalos taškų skirtumas yra mažiausias. Atspausdinkite informaciją apie žaidėją į ekraną. Jei yra keli, spausdinkite visus.
* Sudarykite visų herojų klasių sąrašą, klasių pavadinimus įrašykite į failą „Klasės.csv“ .

## Programos tekstas

//Hero.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace U1\_24\_NR\_ND

{

class Hero

{

public string Name { get; set; }

public string Race { get; set; }

public string Class { get; set; }

public int LifePoints { get; set; }

public int ManaPoints { get; set; }

public int AtkPoints { get; set; }

public int DefPoints { get; set; }

public int StrPoints { get; set; }

public int SpdPoints { get; set; }

public int IntPoints { get; set; }

public string Special { get; set; }

public Hero(string name, string race, string \_class, int lifePoints, int manaPoints, int atkPoints, int defPoints, int strPoints, int spdPoints, int intPoints, string special)

{

this.Name = name;

this.Race = race;

this.Class = \_class;

this.LifePoints = lifePoints;

this.ManaPoints = manaPoints;

this.AtkPoints = atkPoints;

this.DefPoints = defPoints;

this.StrPoints = strPoints;

this.SpdPoints = spdPoints;

this.IntPoints = intPoints;

this.Special = special;

}

}

}

//IOUtils.cs

using System;

using System.IO;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace U1\_24\_NR\_ND

{

static class IOUtils

{

public static List<Hero> ReadHeroes(string fileName)

{

List<Hero> output = new List<Hero>();

string[] lines = new string[100];

// file error handling

if (System.IO.File.Exists(fileName))

{

lines = File.ReadAllLines(fileName, Encoding.UTF8);

}

else

{

Console.WriteLine("Failas nerastas. Programa negali veikti.");

System.Environment.Exit(1); // exit code 1 means that the program did not run successfuly

}

if (lines.Length <= 0)

{

Console.WriteLine("Pateiktas tuščias failas. Programa negali veikti.");

System.Environment.Exit(1); // exit code 1 means that the program did not run successfuly

}

foreach (string line in lines)

{

string[] values = line.Split(';');

string name = values[0];

string race = values[1];

string \_class = values[2];

int lifePoints = int.Parse(values[3]);

int manaPoints = int.Parse(values[4]);

int atkPoints = int.Parse(values[5]);

int defPoints = int.Parse(values[6]);

int strPoints = int.Parse(values[7]);

int spdPoints = int.Parse(values[8]);

int intPoints = int.Parse(values[9]);

string special = values[10];

Hero heroToAdd = new Hero(

name,

race,

\_class,

lifePoints,

manaPoints,

atkPoints,

defPoints,

strPoints,

spdPoints,

intPoints,

special

);

output.Add(heroToAdd);

}

return output;

}

public static void PrintHeroes(List<Hero> input)

{

PrintIndexedTableLine(18, 11, '┌', '┬', '┐', '─');

Console.WriteLine(

"│ {0,-16} │ {1,-16} │ {2,-16} │ {3,-16} │ {4,-16} │ {5, -16} │ {6, -16} │ {7, -16} │ {8, -16} │ {9, -16} │ {10, -16} │",

"Vardas",

"Rasė",

"Klasė",

"Gyvybės t.",

"Manos t.",

"Žalos t.",

"Gynybos t.",

"Jėga",

"Vikrumas",

"Intelektas",

"Ypat. galia"

);

PrintIndexedTableLine(18, 11, '├', '┼', '┤', '─');

for (int i = 0; i < input.Count; i++)

{

Hero hero = input[i];

Console.WriteLine(

"│ {0,-16} │ {1,-16} │ {2,-16} │ {3,-16} │ {4,-16} │ {5, -16} │ {6, -16} │ {7, -16} │ {8, -16} │ {9, -16} │ {10, -16} │",

hero.Name,

hero.Race,

hero.Class,

hero.LifePoints,

hero.ManaPoints,

hero.AtkPoints,

hero.DefPoints,

hero.StrPoints,

hero.SpdPoints,

hero.IntPoints,

Truncate(hero.Special, 12)

);

if (i == input.Count - 1)

{

PrintIndexedTableLine(18, 11, '└', '┴', '┘', '─');

}

else

{

PrintIndexedTableLine(18, 11, '├', '┼', '┤', '─');

}

}

}

// prints out a list of heroes with some of their info not shown

public static void PrintHeroesCompressed(List<Hero> input)

{

PrintIndexedTableLine(18, 4, '┌', '┬', '┐', '─');

Console.WriteLine(

"│ {0,-16} │ {1,-16} │ {2,-16} │ {3,-16} │",

"Vardas",

"Rasė",

"Klasė",

"Gyvybės t."

);

PrintIndexedTableLine(18, 4, '├', '┼', '┤', '─');

for (int i = 0; i < input.Count; i++)

{

Hero hero = input[i];

Console.WriteLine(

"│ {0,-16} │ {1,-16} │ {2,-16} │ {3,-16} │",

hero.Name,

hero.Race,

hero.Class,

hero.LifePoints

);

if (i == input.Count - 1)

{

PrintIndexedTableLine(18, 4, '└', '┴', '┘', '─');

}

else

{

PrintIndexedTableLine(18, 4, '├', '┼', '┤', '─');

}

}

}

// a method to truncate strings that are too long

private static string Truncate(string value, int maxChars)

{

return value.Length <= maxChars ? value : value.Substring(0, maxChars) + "...";

}

// a simple method to assist in creating text character based tables

private static void PrintIndexedTableLine(int spacing, int columnCount, char leftEdge, char middleEdge, char rightEdge, char line)

{

Console.Write(leftEdge);

for (int i = 0; i < columnCount; i++) {

Console.Write(new string(line, spacing));

if (i == columnCount - 1)

{

Console.WriteLine(rightEdge);

}

else

{

Console.Write(middleEdge);

}

}

}

public static void OutputClassesToCSV(string fileName, List<String> classes)

{

string[] lines = classes.ToArray();

File.WriteAllLines(fileName, lines, Encoding.UTF8);

}

}

}

//TaskUtils.cs

using System;

using System.IO;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace U1\_24\_NR\_ND

{

class TaskUtils

{

public static List<Hero> FindHeroesWithHighestHealth(List<Hero> input)

{

List<Hero> output = new List<Hero>();

foreach (Hero hero in input)

{

if (output.Count == 0)

{

output.Add(hero);

continue;

}

Hero heroToCompare = output[0];

if (hero.LifePoints > heroToCompare.LifePoints)

{

output.Clear();

output.Add(hero);

}

else if (hero.LifePoints == heroToCompare.LifePoints)

{

output.Add(hero);

}

}

return output;

}

// a method that finds all heroes with the smallest difference between

// their atkPoints and defPoints

public static List<Hero> FindHeroesWithSmallestDifference(List<Hero> input)

{

List<Hero> output = new List<Hero>();

foreach (Hero hero in input)

{

if (output.Count == 0)

{

output.Add(hero);

continue;

}

Hero heroToCompare = output[0];

int aDiff = Math.Abs(hero.AtkPoints-hero.DefPoints);

int bDiff = Math.Abs(heroToCompare.AtkPoints - heroToCompare.DefPoints);

if (aDiff < bDiff)

{

output.Clear();

output.Add(hero);

}

else if (aDiff == bDiff)

{

output.Add(hero);

}

}

return output;

}

public static List<String> FindUniqueClasses(List<Hero> input)

{

List<String> output = new List<String>();

foreach (Hero hero in input)

{

if (!output.Contains(hero.Class))

{

output.Add(hero.Class);

}

}

return output;

}

}

}

//Program.cs

using System;

using System.IO;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace U1\_24\_NR\_ND

{

class Program

{

public static void Main(string[] args)

{

// read from file

List<Hero> allHeroes = IOUtils.ReadHeroes("herojai.csv");

// print out all heroes

Console.WriteLine("Visi herojai:");

IOUtils.PrintHeroes(allHeroes);

// print out heroes with highest [LifePoints]

Console.WriteLine("Herojai su didžiausiu kiekiu gyvybės taškų:");

IOUtils.PrintHeroesCompressed(TaskUtils.FindHeroesWithHighestHealth(allHeroes));

// print out all heroes with the smallest difference between

// [AtkPoints] and [DefPoints]

Console.WriteLine("Herojai su mažiausiu skirtumu tarp žalos ir gynybos taškų:");

IOUtils.PrintHeroes(TaskUtils.FindHeroesWithSmallestDifference(allHeroes));

List<String> uniqueClasses = TaskUtils.FindUniqueClasses(allHeroes);

IOUtils.OutputClassesToCSV("Klasės.csv", uniqueClasses);

}

}

}

## Pradiniai duomenys ir rezultatai

Pradiniai duomenys:

Aloyzas;Lokys;Kunigas;97;72;38;35;7;7;1;Gerai gamina maistą

Aloyzas;Žmogus;Dainininkas;42;69;82;73;4;5;8;Labai laimingas

Antanas;Varliažmogis;Kunigas;18;27;25;51;1;9;9;Labai gražios akys

Antanas;Varliažmogis;Kunigas;66;87;99;25;4;2;4;Ugnies valdymas

Petras;Driežažmogis;Burtininkas;21;55;20;17;5;1;1;Labai laimingas

Vardėnis;Driežažmogis;Magas;59;33;40;55;0;0;8;Labai gražios akys

Motiejus;Elfas;Lankininkas;35;44;45;40;3;7;5;Labai laimingas

Motiejus;Tamsusis elfas;Kunigas;86;70;26;69;8;9;7;Labai gražios akys

Motiejus;Elfas;Karys;57;71;95;51;4;7;2;Labai laimingas

Antanas;Driežažmogis;Riteris;33;6;91;80;3;4;7;Ugnies valdymas

Vardėnis;Lokys;Gydytojas;18;34;43;12;1;3;5;Ugnies valdymas

Motiejus;Tamsusis elfas;Lankininkas;75;63;18;22;1;2;7;Labai laimingas

Juozas;Šuo;Karys;12;56;86;38;6;7;4;Gerai gamina maistą

Vardėnis;Varliažmogis;Burtininkas;37;47;14;75;1;6;6;Gerai gamina maistą

Motiejus;Žmogus;Riteris;28;23;61;81;3;1;9;Labai gražios akys

Antanas;Elfas;Riteris;56;4;16;91;9;7;5;Labai laimingas

Juozas;Žmogus;Karys;97;17;74;69;5;7;5;Ugnies valdymas

Motiejus;Varliažmogis;Magas;39;74;21;31;7;5;0;Labai gražios akys

Petras;Žmogus;Riteris;46;64;92;83;8;6;9;Labai laimingas

Vardėnis;Elfas;Karys;61;77;81;26;1;1;9;Gerai gamina maistą

Petras;Driežažmogis;Gydytojas;92;35;99;37;3;4;0;Labai gražios akys

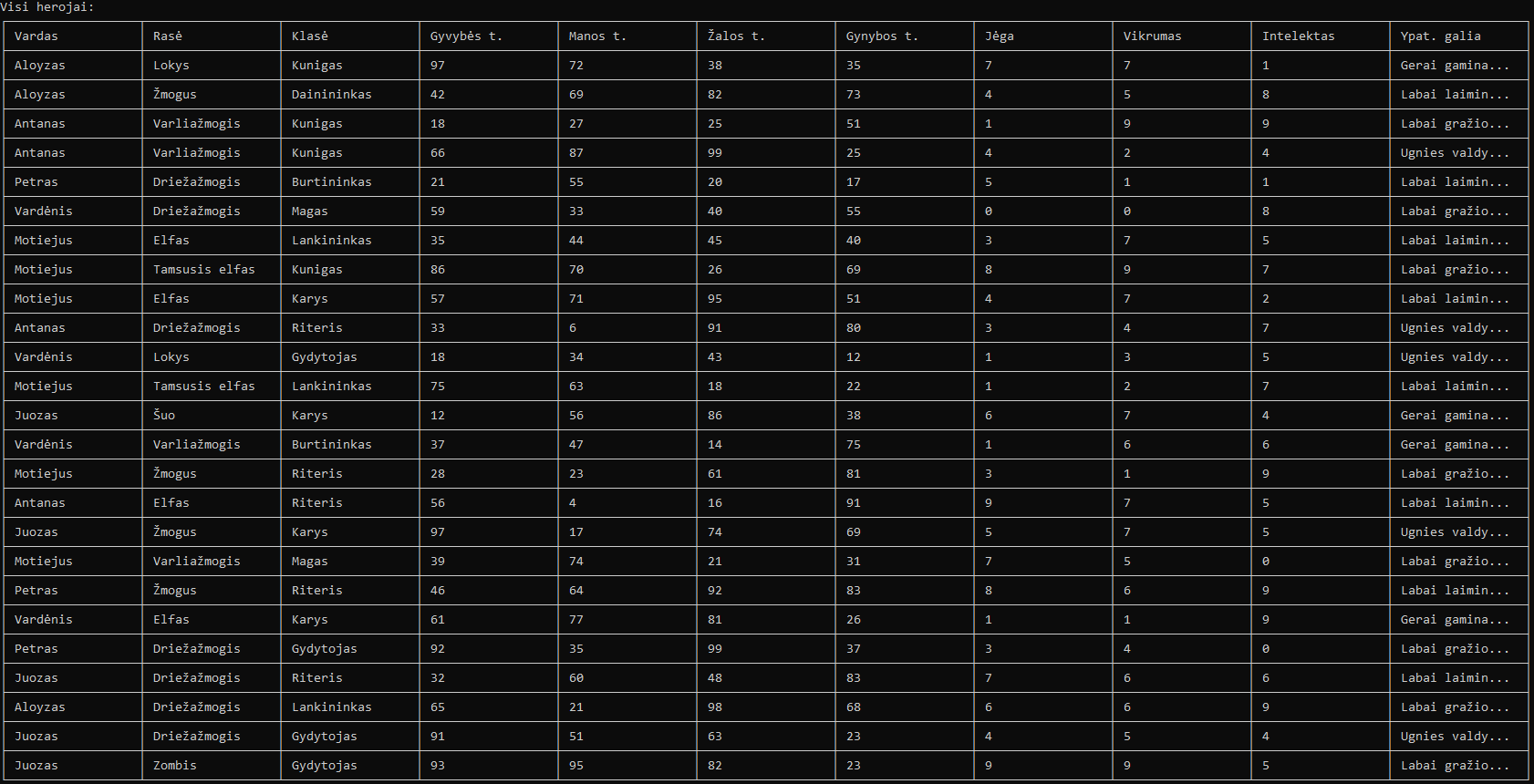
Juozas;Driežažmogis;Riteris;32;60;48;83;7;6;6;Labai laimingas

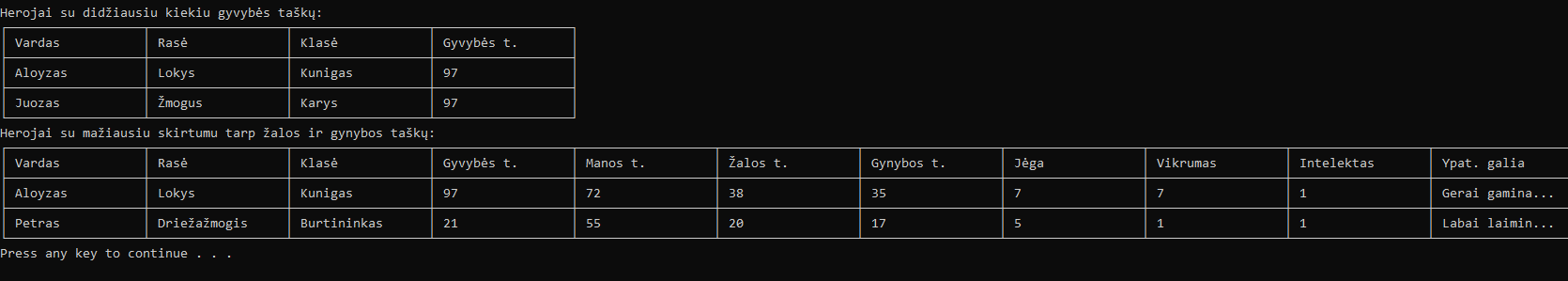
Aloyzas;Driežažmogis;Lankininkas;65;21;98;68;6;6;9;Labai gražios akys

Juozas;Driežažmogis;Gydytojas;91;51;63;23;4;5;4;Ugnies valdymas

Juozas;Zombis;Gydytojas;93;95;82;23;9;9;5;Labai gražios akys

Rezultatai:





Klasės.csv:

Kunigas

Dainininkas

Burtininkas

Magas

Lankininkas

Karys

Riteris

Gydytojas

## Dėstytojo pastabos

# Skaičiavimų klasė

## Darbo užduotis

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos

# Konteineris

## Darbo užduotis

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos

# Teksto analizė ir redagavimas

## Darbo užduotis

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos

# Paveldėjimas

## Darbo užduotis

## Programos tekstas

## Pradiniai duomenys ir rezultatai

## Dėstytojo pastabos