5. Paveldėjimas

5.1. Darbo užduotis

U5_1. Krepšinio rinktinė. Turite trijų pastarųjų metų į stovyklas pakviestų krepšininkų sąrašus. Pirmoje eilutėje – metai, antroje – stovyklos pradžios data, trečioje – stovyklos pabaigos data. Toliau informacija apie rinktinės narius. Krepšinio rinktinę sudaro ne tik krepšininkai, bet ir pagalbinis personalas – treneriai, gydytojai, masažuotojai ir kt. Sukurkite klasę „Member“ (savybės – vardas, pavardė, gimimo data), kurią paveldės klasės „Player“ (savybės – ūgis, pozicija, klubas, požymis „pakviestas“, požymis „kapitonas“) ir „Staff“ (savybė – pareigos).

- Sudarykite rinktinės narių (tiek krepšininkų, tiek personalo), dalyvavusių visose trijose stovyklose, sąrašą ir atspausdinkite jį ekrane.
- Sudarykite visų puolėjų, dalyvavusių rinktinės stovyklose, sąrašą ir ekrane atspausdinkite jų vardus, pavardes bei ūgį. Sudarykite vyr. trenerių sąrašą ir atspausdinkite ekrane visų jų vardus ir pavardes.
- Atspausdinkite kiekvienos stovyklos dalyvių sąrašą, išrikiuojant juos pagal amžių nuo vyriausio iki jauniausio.
- Sudarykite visų į rinktinę pakviestų krepšininkų sąrašą. Jų duomenis įrašykite į failą „Rinktinė.csv“. Sudarykite viso į rinktinę pakviesto personalo sąrašą. Jų duomenis įrašykite į failą „Personalas.csv“.

5.2. Programos tekstas

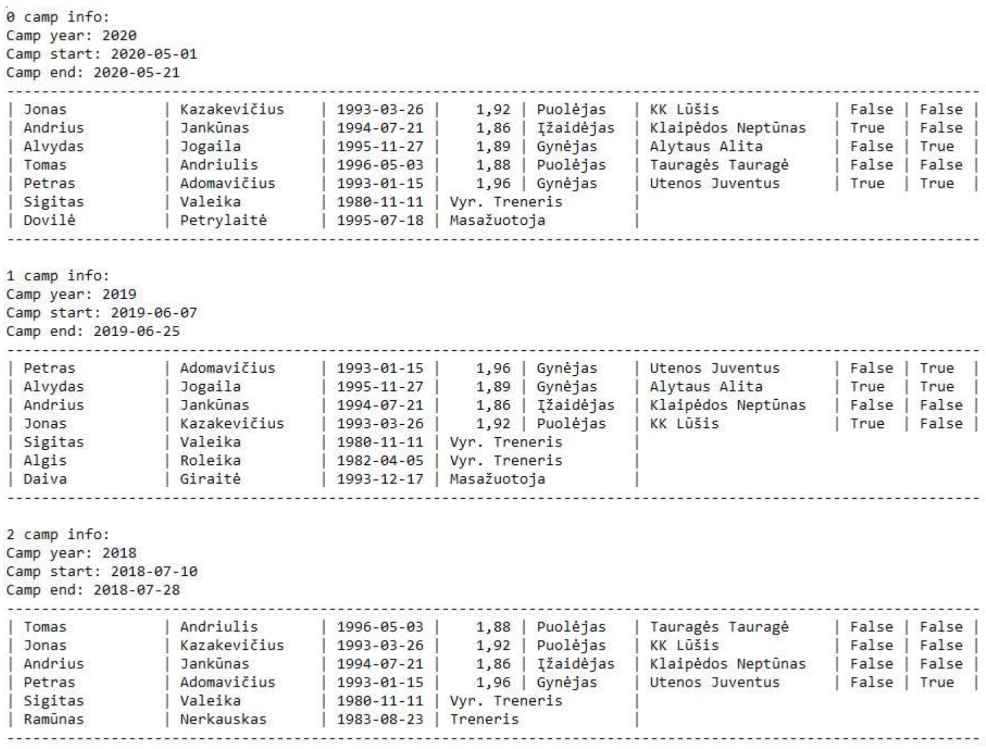
```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;

class Branch

    {
        public int Year { get; set; }
        public DateTime StartDate { get; set; }
        public DateTime EndDate { get; set; }

        public MemberContainer Players { get; set; }
        public MemberContainer Staff { get; set; }

        public Branch (int year, DateTime startDate, DateTime endDate)
        {
            Year = year;
            StartDate = startDate;
            EndDate = endDate;
            Players = new MemberContainer(Program.MaxNumberOfMembers);
            Staff = new MemberContainer(Program.MaxNumberOfMembers);
        }

        public void AddPlayer(Player player)
        {
            Players.AddMember(player);
        }

        public void AddStaff(Staff staff)
        {
            Staff.AddMember(staff);
        }

        /// <summary>
        /// Returns all members from one branch in one container
        /// </summary>
        /// <returns></returns>
        public MemberContainer GetAllMembers()
        {
            MemberContainer allMembers = new 
MemberContainer(Program.MaxNumberOfMembers);

            for (int i = 0; i < Players.Count; i++)
            {
                allMembers.AddMember(Players.GetMember(i));
            }

            for (int j = 0; j < Staff.Count; j++)
            {
                allMembers.AddMember(Staff.GetMember(j));
            }

            return allMembers;
        }
    }

class InOut
    {
        /// <summary>
        /// Calling method for main data reading method
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        public static void ReadFiles(Branch[] branches, ref int number)
        {
            string[] fileNames = Directory.GetFiles(Directory.GetCurrentDirectory(), 
"*.csv");
            foreach (var file in fileNames)
            {
                ReadMemberData(file, branches, ref number);
            }
        }

        /// <summary>
        /// Reads all data from a branch
        /// </summary>
        /// <param name="fileName"></param>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        private static void ReadMemberData(string fileName, Branch[] branches, ref int 
number)
        {
            string[] lines = File.ReadAllLines(fileName);
            int year = int.Parse(lines[0]);
            DateTime startDate = DateTime.Parse(lines[1]);
            DateTime endDate = DateTime.Parse(lines[2]);
            Branch branch = TaskUtils.GetBranch(branches, ref number, year, 
startDate,endDate);
            for (int i = 3; i < lines.Length; i++)
            {
                string[] values = lines[i].Split(';');
                char type = char.Parse(values[0]);
                string firstName = values[1];
                string lastName = values[2];
                DateTime birthDate = DateTime.Parse(values[3]);
                switch (type)
                {
                    case 'P':
                        double height = double.Parse(values[4]);
                        string position = values[5];
                        string club = values[6];
                        bool isInvited = bool.Parse(values[7]);
                        bool isCaptain = bool.Parse(values[8]);

                        Player player = new Player(firstName, lastName, birthDate,
height, position, club, isInvited, isCaptain);
                        if (!branch.Players.Contains(player))
                            branch.AddPlayer(player);
                        break;

                    case 'S':
                        string duty = values[4];
                        Staff staff = new Staff(firstName, lastName, birthDate, duty);
                        if (!branch.Staff.Contains(staff))
                            branch.AddStaff(staff);
                        break;
                }
            }
        }

        /// <summary>
        /// Prints members to screen with all of their info
        /// </summary>
        /// <param name="members"></param>
        /// <param name="header"></param>
        public static void PrintMembersToScreen(MemberContainer members, string header)
        {
            Console.WriteLine("\n" + header);

            Console.WriteLine(new string('-', 112));

            if (members.Count == 0)
                Console.WriteLine("Narių nėra");
            else
            {
                for (int i = 0; i < members.Count; i++)
                {
                Console.WriteLine(members.GetMember(i).ToString());
                }
            }
            Console.WriteLine(new string('-', 112));
        }

        /// <summary>
        /// Calling method for printing each branch members
        /// </summary>
        /// <param name="membersFromBranch"></param>
        /// <param name="header"></param>
        public static void PrintMembersFromBranches(MemberContainer[]
membersFromBranch,string header)
        {
            for (int i = 0; i < membersFromBranch.Length; i++)
            {
                PrintMembersToScreen(membersFromBranch[i], i+1 + "-os stovyklos
dalyviai:");
            }
        }
        /// <summary>
        /// Prints players to screen as name,last name and height
        /// </summary>
        /// <param name="players"></param>
        /// <param name="header"></param>
        public static void PrintPlayersToScreen(List<Player> players, string header)
        {

            Console.WriteLine("\n" + header);
            Console.WriteLine(new string('-', 40));

            if (players.Count == 0)
                Console.WriteLine("Narių nėra");
            else
            {
                Console.WriteLine("| {0,-10} | {1,-15} | {2,5} |", "Vardas", "Pavardė", 
"Ūgis");
                Console.WriteLine(new string('-', 40));

                for (int i = 0; i < players.Count; i++)
                {
                    Player player = players[i];
                    Console.WriteLine(string.Format("| {0,-10} | {1,-15} | {2,5} |", 
player.FirstName, player.LastName, player.Height));
                }
            }
            Console.WriteLine(new string('-', 40));
        }
        /// <summary>
        /// Prints only members first and last names to screen
        /// </summary>
        /// <param name="members"></param>
        /// <param name="header"></param>
        public static void PrintMemberNamesToScreen(MemberContainer members,string 
header)
        {
            Console.WriteLine("\n" + header);

            Console.WriteLine(new string('-', 32));

            if(members.Count == 0)
                Console.WriteLine("Narių nėra");
            else
            {
                for (int i = 0; i < members.Count; i++)
                {
                    Member member = members.GetMember(i);
                    Console.WriteLine(string.Format("| {0,-10} | {1,-15} |", 
member.FirstName, member.LastName));
                }
            }
            Console.WriteLine(new string('-', 32));
        }


        /// <summary>
        /// Prints all read data to .txt file
        /// </summary>
        /// <param name="fileName"></param>
        /// <param name="branches"></param>
        public static void PrintDataToTXT(string fileName, Branch[] branches)
        {
            List<string> lines = new List<string>();
            for (int i = 0; i < branches.Length; i++)
            {
                lines.Add(String.Format("\n" + i + " camp info:"));

                lines.Add(String.Format("Camp year: " + branches[i].Year));
                lines.Add(String.Format("Camp start: " +
branches[i].StartDate.ToShortDateString()));
                lines.Add(String.Format("Camp end: " +
branches[i].EndDate.ToShortDateString()));

                lines.Add(String.Format(new string('-', 112)));

                MemberContainer members = branches[i].GetAllMembers();

                for (int j = 0; j < members.Count; j++)
                {
                    lines.Add(members.GetMember(j).ToString());
                }
                lines.Add(String.Format(new string('-', 112)));
            }
            File.AppendAllLines(fileName, lines, Encoding.UTF8);
        }

        /// <summary>
        /// Prints members to .csv file
        /// </summary>
        /// <param name="fileName"></param>
        /// <param name="members"></param>
        /// <param name="header"></param>
        public static void PrintPlayersToCSV(string fileName, MemberContainer members,
string header)
        {
            List<string> lines = new List<string>();

            lines.Add("\n" + header);

            if(members.Count == 0)
            {
                lines.Add("Narių nėra");
            }
            else
            {
                for (int i = 0; i < members.Count; i++)
                {
                    Player player = members.GetMember(i) as Player;
                    lines.Add(String.Format("{0};{1};{2};{3};{4};{5};{6};{7}",
player.FirstName, player.LastName, player.BirthDate.ToShortDateString(), player.Height,
player.Position,player.Club, player.IsInvited, player.IsCaptain));
                }
            }

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

abstract class Member
    {
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public DateTime BirthDate { get; set; }

        public Member(string firstName, string lastName, DateTime birthDate)
        {
            FirstName = firstName;
            LastName = lastName;
            BirthDate = birthDate;
        }

        public override bool Equals(object obj)
        {
            Member other = obj as Member;
            return other != null && other.FirstName == FirstName && other.LastName ==
LastName && other.BirthDate == BirthDate;
        }

        /// <summary>
        /// Compares members by their birthdates
        /// </summary>
        /// <param name="other"></param>
        /// <returns></returns>
        public int CompareTo(Member other)
        {
            if (this.BirthDate.CompareTo(other.BirthDate) > 0)
                return 1;
            if (this.BirthDate.CompareTo(other.BirthDate) < 0)
                return 0;
            return 0;
        }

        public override string ToString()
        {
            return string.Format("| {0,-10} | {1,-15} | {2,10} |", FirstName, LastName,
BirthDate.ToShortDateString());
        }
    }

class MemberContainer
    {
        private Member[] Members;
        public int Count { get; private set; }

        public MemberContainer(int size)
        {
            Members = new Member[size];
        }

        public MemberContainer()
        {
        }

        public void AddMember(Member member)
        {
            Members[Count] = member;
            Count++;
        }

        public void SetMember(int index, Member member)
        {
            Members[index] = member;
        }

        public Member GetMember(int index)
        {
            return Members[index];
        }

        /// <summary>
        /// Returns index of given member
/// </summary>
/// <param name="member"></param>
/// <returns></returns>
public int GetIndex(Member member)
{
    int index = 0;
    for (int i = 0; i < Members.Count(); i++)
    {
        if (member.Equals(Members[i]))
        {
            index = i;
            break;
        }
    }
    return index;
}

/// <summary>
/// Removes member from member container
/// </summary>
/// <param name="member"></param>
public void RemoveMember(Member member)
{
    int i = 0;
    while (i < Count)
    {
        if (Members[i].Equals(member))
        {
            Count--;
            for (int j = i; j < Count; j++)
            {
                Members[j] = Members[j + 1];
            }
            break;
        }
        i++;
    }
}

public bool Contains(Member member)
{
    return Members.Contains(member);
}

/// <summary>
/// Sorts members by overided CompareTo method
/// </summary>
public void Sort()
{
    Member member = this.Members[0];
    int m;

    for (int i = 0; i < this.Count; i++)
    {
        m = i;
        for (int j = 0; j < this.Count; j++)
        {
            if (this.Members[j].CompareTo(this.Members[i]) > 0)
            {
                m = j;
                member = this.Members[i];
                this.Members[i] = this.Members[m];
                this.Members[m] = member;
            }
```

```
                }
            }
        }
    }

class Player : Member
    {
        public double Height { get; set; }
        public string Position { get; set; }
        public string Club { get; set; }
        public bool IsInvited { get; set; }
        public bool IsCaptain { get; set; }

        public Player(string firstName, string lastName, DateTime birthDate,
            double height, string position, string club, bool isInvited, bool
isCaptain)
            : base(firstName,lastName,birthDate)
        {
            Height = height;
            Position = position;
            Club = club;
            IsInvited = isInvited;
            IsCaptain = isCaptain;
        }

        public override string ToString()
        {
            return string.Format("| {0,-15} | {1,-15} | {2,-10} | {3,7} | {4,-10} |
{5,-20} | {6,-5} | {7,-5} |",
                FirstName, LastName, BirthDate.ToShortDateString(), Height, Position,
Club, IsInvited, IsCaptain);
        }
    }

class Staff : Member
    {
        public string Duty { get; set; }

        public Staff(string firstName, string lastName, DateTime birthDate, string
duty)
            : base(firstName, lastName, birthDate)
        {
            Duty = duty;
        }

        public override string ToString()
        {
            return string.Format("| {0,-15} | {1,-15} | {2,-10} | {3,-20} |",
                FirstName, LastName, BirthDate.ToShortDateString(),Duty);
        }
    }

class TaskUtils
    {
        /// <summary>
        /// Returns branch from given branch info
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        /// <param name="year"></param>
        /// <param name="startDate"></param>
        /// <param name="endDate"></param>
        /// <returns></returns>
        public static Branch GetBranch(Branch[] branches, ref int number,int year,
DateTime startDate, DateTime endDate)
        {
            for (int i = 0; i < number; i++)
            {
                if (branches[i].Year == year && branches[i].StartDate == startDate &&
branches[i].EndDate == endDate)
                {
                    return branches[i];
                }
            }
            branches[number++] = new Branch(year, startDate, endDate);
            return branches[number - 1];
        }

        /// <summary>
        /// Returns members that are in all branches
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="branchCount"></param>
        /// <returns></returns>
        public static MemberContainer AttendedAll (Branch[] branches, int branchCount)
        {
            MemberContainer AttendedAll = new MemberContainer();

            for (int i = 0; i < branchCount; i++)
            {
                if (i == 0)
                    AttendedAll = branches[i].GetAllMembers();
                else
                    AttendedAll = RemoveUnattended(AttendedAll, branches[i]);
            }
            return AttendedAll;
        }

        /// <summary>
        /// Removes members that aren't in all branches
        /// </summary>
        /// <param name="Attendees"></param>
        /// <param name="branch"></param>
        /// <returns></returns>
        public static MemberContainer RemoveUnattended(MemberContainer Attendees,
Branch branch)
        {
            MemberContainer branchMembers = branch.GetAllMembers();

            for (int i = 0; i < Attendees.Count; i++)
            {
                if (!branchMembers.Contains(Attendees.GetMember(i)))
                    Attendees.RemoveMember(Attendees.GetMember(i));
            }
            return Attendees;
        }

        /// <summary>
        /// Returns all players from branches
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        /// <returns></returns>
        public static MemberContainer AllPlayersContainer(Branch[] branches,int number)
        {
            MemberContainer allPlayers = new MemberContainer(Program.MaxNumberOfMembers
* Program.MaxNumberOfBranches);
```

```
            for (int i = 0; i < number; i++)
            {
                for (int j = 0; j < branches[i].Players.Count; j++)
                {
                    Player player = branches[i].Players.GetMember(j) as Player;
                    if(!allPlayers.Contains(player))
                    {
                        allPlayers.AddMember(player);
                    }
                    else
                    if(allPlayers.Contains(player) && player.IsInvited == true)
                    {
                        allPlayers.SetMember(allPlayers.GetIndex(player), player);
                    }
                }
            }
            return allPlayers;
        }

        /// <summary>
        /// Returns all staff from branches
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        /// <returns></returns>
        public static MemberContainer AllStaff(Branch[] branches, int number)
        {
            MemberContainer allStaff = new MemberContainer(Program.MaxNumberOfMembers *
Program.MaxNumberOfBranches);
            for (int i = 0; i < number; i++)
            {
                for (int j = 0; j < branches[i].Staff.Count; j++)
                {
                    if(!allStaff.Contains(branches[i].Staff.GetMember(j)))
                    allStaff.AddMember(branches[i].Staff.GetMember(j));
                }
            }
            return allStaff;
        }

        /// <summary>
        /// Returns players that are in position of attacker
        /// </summary>
        /// <param name="players"></param>
        /// <returns></returns>
        public static List<Player> GetAttackers(MemberContainer players)
        {
            List<Player> OnlyAttackers = new List<Player>();
            for (int i = 0; i < players.Count; i++)
            {
                Player player = players.GetMember(i) as Player;

                if (player.Position == "Puolėjas")
                    OnlyAttackers.Add(player);
            }

            return OnlyAttackers;
        }

        /// <summary>
        /// Returns all headtrainers from all staff
        /// </summary>
        /// <param name="staff"></param>
        /// <returns></returns>
        public static MemberContainer GetHeadTrainers(MemberContainer staff)
        {
            MemberContainer headTrainers = new
MemberContainer(Program.MaxNumberOfMembers * Program.MaxNumberOfBranches);
            for (int i = 0; i < staff.Count; i++)
            {
                Staff staffMember = staff.GetMember(i) as Staff;

                if (staffMember.Duty == "Vyr. Treneris" &&
!headTrainers.Contains(staff.GetMember(i)))
                {
                    headTrainers.AddMember(staff.GetMember(i));
                }
            }

            return headTrainers;
        }

        /// <summary>
        /// Returns all members from each branch in separate containers
        /// </summary>
        /// <param name="branches"></param>
        /// <param name="number"></param>
        /// <returns></returns>
        public static MemberContainer[] GetEachBranchMembers (Branch[] branches,int
number)
        {
            MemberContainer[] branchMemberArray = new MemberContainer[number];

            for (int i = 0; i < number; i++)
            {
                branchMemberArray[i] = branches[i].GetAllMembers();
            }

            return branchMemberArray;
        }

        /// <summary>
        /// Sorts the container array
        /// </summary>
        /// <param name="memberContainers"></param>
        /// <returns></returns>
        public static MemberContainer[] SortEachContainer (MemberContainer[]
memberContainers)
        {
            for (int i = 0; i < memberContainers.Length; i++)
            {
                memberContainers[i].Sort();
            }

            return memberContainers;
        }

        /// <summary>
        /// Returns only players that are invited
        /// </summary>
        /// <param name="players"></param>
        /// <returns></returns>
        public static MemberContainer GetInvitedPlayers(MemberContainer players)
        {
            MemberContainer invitedOnly = new
MemberContainer(Program.MaxNumberOfBranches*Program.MaxNumberOfMembers);

            for (int i = 0; i < players.Count; i++)
            {
                Player player = players.GetMember(i) as Player;
                if(player.IsInvited)
                {
                    invitedOnly.AddMember(players.GetMember(i));
                }
            }
            return invitedOnly;
        }
    }

class Program
    {
        public const int MaxNumberOfBranches = 3;
        public const int MaxNumberOfMembers = 30;

        static void Main(string[] args)
        {
            string Sdata = "StartingData.txt";

            File.Delete("Rinktine.csv");

            Branch[] branches = new Branch[MaxNumberOfBranches];
            int branchCount = 0;

            InOut.ReadFiles(branches, ref branchCount);

            File.Delete(Sdata);
            InOut.PrintDataToTXT(Sdata, branches);

            MemberContainer allPlayers = TaskUtils.AllPlayersContainer(branches, 
branchCount);
            MemberContainer allStaff = TaskUtils.AllStaff(branches, branchCount);

            MemberContainer AttendedAll = TaskUtils.AttendedAll(branches, branchCount);
            InOut.PrintMembersToScreen(AttendedAll, "Nariai dalyvavę visose 
stovyklose:");

            List<Player> allAttackers = TaskUtils.GetAttackers(allPlayers);
            InOut.PrintPlayersToScreen(allAttackers, "Visi dalyvavę puolėjai:");

            MemberContainer HeadTrainers = TaskUtils.GetHeadTrainers(allStaff);
            InOut.PrintMemberNamesToScreen(HeadTrainers, "Vyr. treneriai:");

            MemberContainer[] MembersInBranchArray = 
TaskUtils.GetEachBranchMembers(branches, branchCount);
            MembersInBranchArray = TaskUtils.SortEachContainer(MembersInBranchArray);
            InOut.PrintMembersFromBranches(MembersInBranchArray, "Kiekvienos stovyklos 
dalyviai:");

            MemberContainer InvitedPlayers = TaskUtils.GetInvitedPlayers(allPlayers);
            InOut.PrintMembersToCSV("Rinktine.csv", InvitedPlayers, "Žaidėjai pakviesti 
į rinktinę:");

        }
    }
```

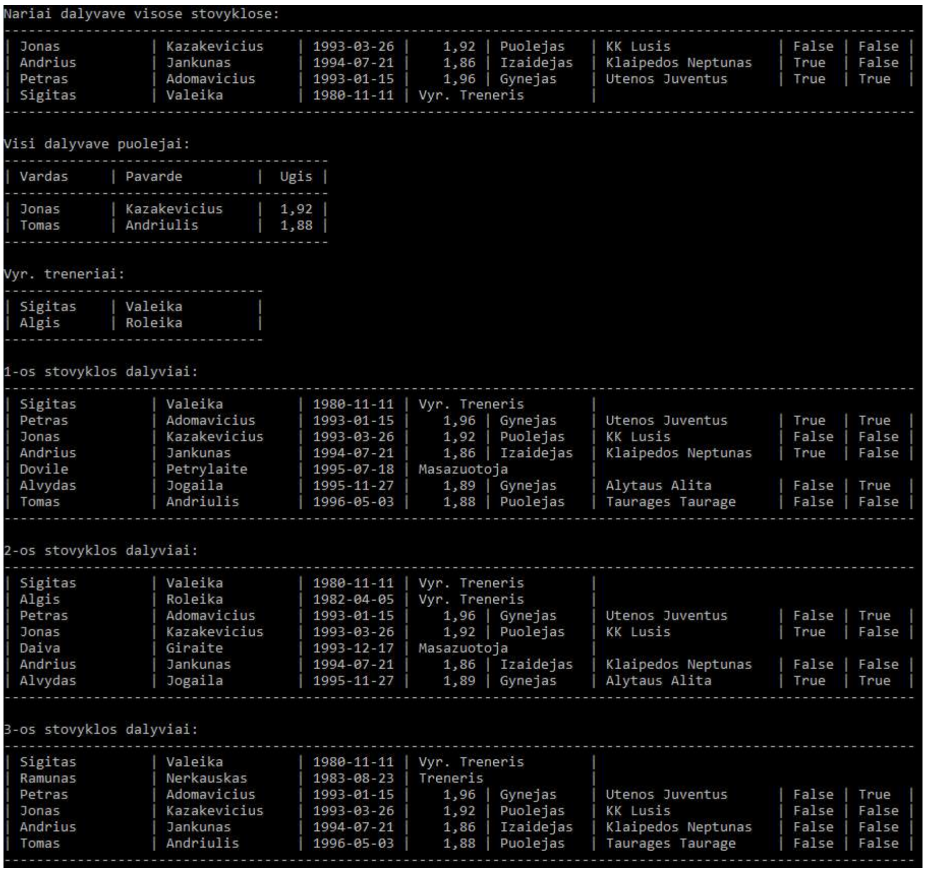
5.3. Pradiniai duomenys ir rezultatai

`StartingData.txt(1):`



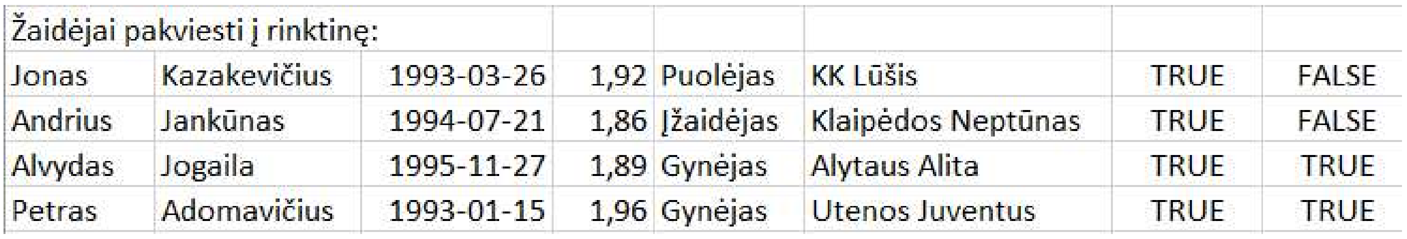
```
0 camp info:
Camp year: 2020
Camp start: 2020-05-01
Camp end: 2020-05-21
-------------------------------------------------------------------------------------------------------------------------
| Jonas          | Kazakevičius    | 1993-03-26 |      1,92 | Puolėjas    | KK Lūšis                 | False | False |
| Andrius        | Jankūnas        | 1994-07-21 |      1,86 | Įžaidėjas   | Klaipėdos Neptūnas       | True  | False |
| Alvydas        | Jogaila         | 1995-11-27 |      1,89 | Gynėjas     | Alytaus Alita            | False | True  |
| Tomas          | Andriulis       | 1996-05-03 |      1,88 | Puolėjas    | Tauragės Tauragė         | False | False |
| Petras         | Adomavičius     | 1993-01-15 |      1,96 | Gynėjas     | Utenos Juventus          | True  | True  |
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Dovilė         | Petrylaitė      | 1995-07-18 | Masažuotoja             |
-------------------------------------------------------------------------------------------------------------------------

1 camp info:
Camp year: 2019
Camp start: 2019-06-07
Camp end: 2019-06-25
-------------------------------------------------------------------------------------------------------------------------
| Petras         | Adomavičius     | 1993-01-15 |      1,96 | Gynėjas     | Utenos Juventus          | False | True  |
| Alvydas        | Jogaila         | 1995-11-27 |      1,89 | Gynėjas     | Alytaus Alita            | True  | True  |
| Andrius        | Jankūnas        | 1994-07-21 |      1,86 | Įžaidėjas   | Klaipėdos Neptūnas       | False | False |
| Jonas          | Kazakevičius    | 1993-03-26 |      1,92 | Puolėjas    | KK Lūšis                 | True  | False |
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Algis          | Roleika         | 1982-04-05 | Vyr. Treneris           |
| Daiva          | Giraitė         | 1993-12-17 | Masažuotoja             |
-------------------------------------------------------------------------------------------------------------------------

2 camp info:
Camp year: 2018
Camp start: 2018-07-10
Camp end: 2018-07-28
-------------------------------------------------------------------------------------------------------------------------
| Tomas          | Andriulis       | 1996-05-03 |      1,88 | Puolėjas    | Tauragės Tauragė         | False | False |
| Jonas          | Kazakevičius    | 1993-03-26 |      1,92 | Puolėjas    | KK Lūšis                 | False | False |
| Andrius        | Jankūnas        | 1994-07-21 |      1,86 | Įžaidėjas   | Klaipėdos Neptūnas       | False | False |
| Petras         | Adomavičius     | 1993-01-15 |      1,96 | Gynėjas     | Utenos Juventus          | False | True  |
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Ramūnas        | Nerkauskas      | 1983-08-23 | Treneris                |
-------------------------------------------------------------------------------------------------------------------------
```

`Console(1):`



```
Nariai dalyvave visose stovyklose:
-------------------------------------------------------------------------------------------------------------------
| Jonas          | Kazakevicius    | 1993-03-26 |      1,92 | Puolejas    | KK Lusis                | False | False |
| Andrius        | Jankunas        | 1994-07-21 |      1,86 | Izaidejas   | Klaipedos Neptunas      | True  | False |
| Petras         | Adomavicius     | 1993-01-15 |      1,96 | Gynejas     | Utenos Juventus         | True  | True  |
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
-------------------------------------------------------------------------------------------------------------------

Visi dalyvave puolejai:
------------------------------------------
| Vardas     | Pavarde          |   Ugis |
------------------------------------------
| Jonas      | Kazakevicius     |   1,92 |
| Tomas      | Andriulis        |   1,88 |
------------------------------------------

Vyr. treneriai:
----------------------------------
| Sigitas    | Valeika          |
| Algis      | Roleika          |
----------------------------------

1-os stovyklos dalyviai:
-------------------------------------------------------------------------------------------------------------------
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Petras         | Adomavicius     | 1993-01-15 |      1,96 | Gynejas     | Utenos Juventus         | True  | True  |
| Jonas          | Kazakevicius    | 1993-03-26 |      1,92 | Puolejas    | KK Lusis                | False | False |
| Andrius        | Jankunas        | 1994-07-21 |      1,86 | Izaidejas   | Klaipedos Neptunas      | True  | False |
| Dovile         | Petrylaite      | 1995-07-18 | Masazuotoja             |
| Alvydas        | Jogaila         | 1995-11-27 |      1,89 | Gynejas     | Alytaus Alita           | False | True  |
| Tomas          | Andriulis       | 1996-05-03 |      1,88 | Puolejas    | Taurages Taurage        | False | False |
-------------------------------------------------------------------------------------------------------------------

2-os stovyklos dalyviai:
-------------------------------------------------------------------------------------------------------------------
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Algis          | Roleika         | 1982-04-05 | Vyr. Treneris           |
| Petras         | Adomavicius     | 1993-01-15 |      1,96 | Gynejas     | Utenos Juventus         | False | True  |
| Jonas          | Kazakevicius    | 1993-03-26 |      1,92 | Puolejas    | KK Lusis                | True  | False |
| Daiva          | Giraite         | 1993-12-17 | Masazuotoja             |
| Andrius        | Jankunas        | 1994-07-21 |      1,86 | Izaidejas   | Klaipedos Neptunas      | False | False |
| Alvydas        | Jogaila         | 1995-11-27 |      1,89 | Gynejas     | Alytaus Alita           | True  | True  |
-------------------------------------------------------------------------------------------------------------------

3-os stovyklos dalyviai:
-------------------------------------------------------------------------------------------------------------------
| Sigitas        | Valeika         | 1980-11-11 | Vyr. Treneris           |
| Ramunas        | Nerkauskas      | 1983-08-23 | Treneris                |
| Petras         | Adomavicius     | 1993-01-15 |      1,96 | Gynejas     | Utenos Juventus         | False | True  |
| Jonas          | Kazakevicius    | 1993-03-26 |      1,92 | Puolejas    | KK Lusis                | False | False |
| Andrius        | Jankunas        | 1994-07-21 |      1,86 | Izaidejas   | Klaipedos Neptunas      | False | False |
| Tomas          | Andriulis       | 1996-05-03 |      1,88 | Puolejas    | Taurages Taurage        | False | False |
-------------------------------------------------------------------------------------------------------------------
```

Rinktine.csv(1):

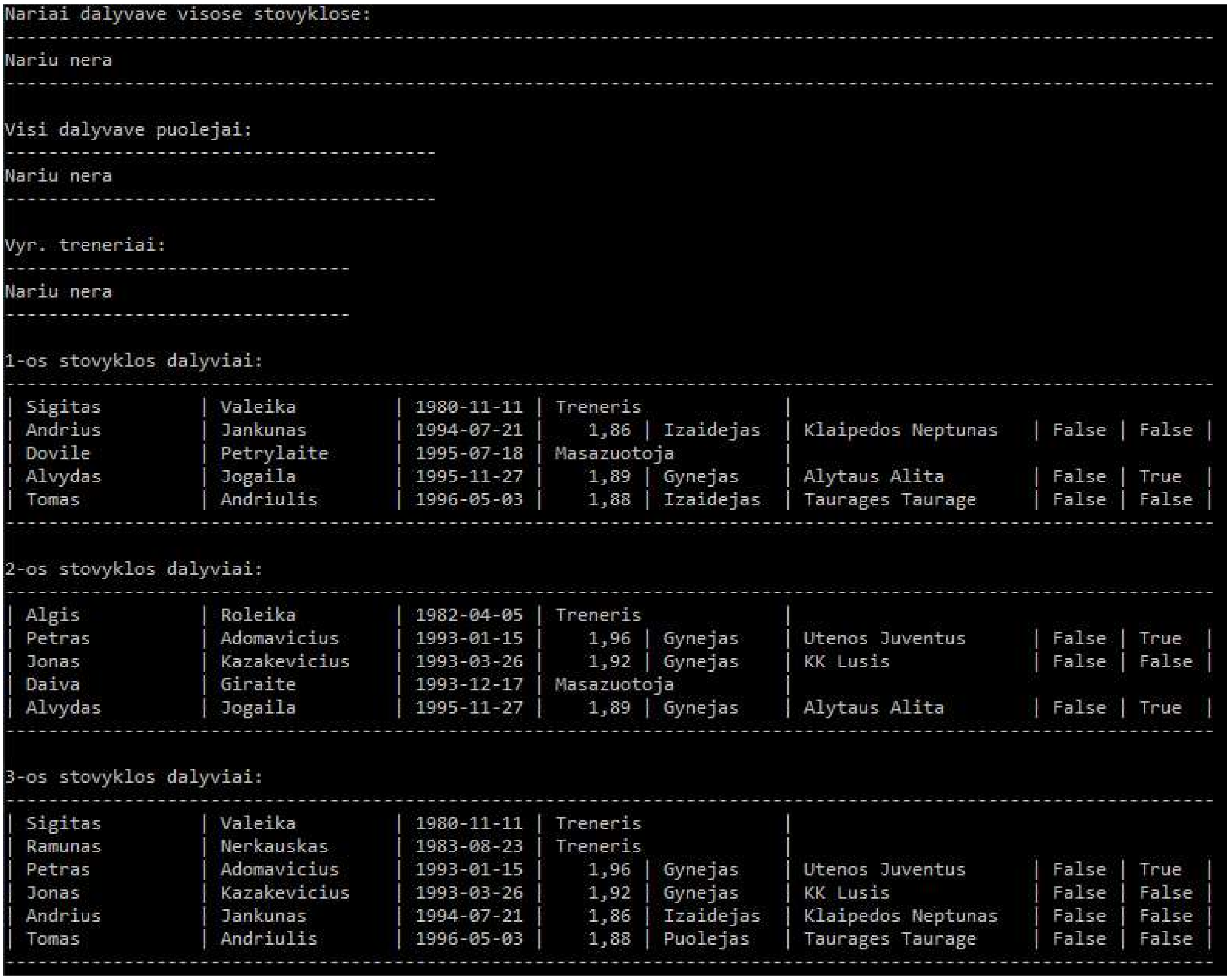


Žaidėjai pakviesti į rinktinę:							
Jonas	Kazakevičius	1993-03-26	1,92	Puolėjas	KK Lūšis	TRUE	FALSE
Andrius	Jankūnas	1994-07-21	1,86	Įžaidėjas	Klaipėdos Neptūnas	TRUE	FALSE
Alvydas	Jogaila	1995-11-27	1,89	Gynėjas	Alytaus Alita	TRUE	TRUE
Petras	Adomavičius	1993-01-15	1,96	Gynėjas	Utenos Juventus	TRUE	TRUE

StartingData.txt(2)

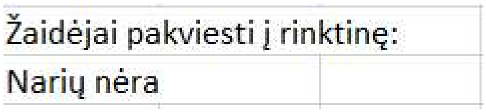
```
0 camp info:
Camp year: 2020
Camp start: 2020-05-01
Camp end: 2020-05-21
------------------------------------------------------------------------------------------------------------------
| Andrius         | Jankūnas        | 1994-07-21 |      1,86 | Įžaidėjas   | Klaipėdos Neptūnas     | False | False |
| Alvydas         | Jogaila         | 1995-11-27 |      1,89 | Gynėjas     | Alytaus Alita          | False | True  |
| Tomas           | Andriulis       | 1996-05-03 |      1,88 | Įžaidėjas   | Tauragės Tauragė       | False | False |
| Sigitas         | Valeika         | 1980-11-11 | Treneris                |
| Dovilė          | Petrylaitė      | 1995-07-18 | Masažuotoja             |
------------------------------------------------------------------------------------------------------------------

1 camp info:
Camp year: 2019
Camp start: 2019-06-07
Camp end: 2019-06-25
------------------------------------------------------------------------------------------------------------------
| Alvydas         | Jogaila         | 1995-11-27 |      1,89 | Gynėjas     | Alytaus Alita          | False | True  |
| Jonas           | Kazakevičius    | 1993-03-26 |      1,92 | Gynėjas     | KK Lūšis               | False | False |
| Petras          | Adomavičius     | 1993-01-15 |      1,96 | Gynėjas     | Utenos Juventus        | False | True  |
| Algis           | Roleika         | 1982-04-05 | Treneris                |
| Daiva           | Giraitė         | 1993-12-17 | Masažuotoja             |
------------------------------------------------------------------------------------------------------------------

2 camp info:
Camp year: 2018
Camp start: 2018-07-10
Camp end: 2018-07-28
------------------------------------------------------------------------------------------------------------------
| Tomas           | Andriulis       | 1996-05-03 |      1,88 | Puolėjas    | Tauragės Tauragė       | False | False |
| Jonas           | Kazakevičius    | 1993-03-26 |      1,92 | Gynėjas     | KK Lūšis               | False | False |
| Andrius         | Jankūnas        | 1994-07-21 |      1,86 | Įžaidėjas   | Klaipėdos Neptūnas     | False | False |
| Petras          | Adomavičius     | 1993-01-15 |      1,96 | Gynėjas     | Utenos Juventus        | False | True  |
| Sigitas         | Valeika         | 1980-11-11 | Treneris                |
| Ramūnas         | Nerkauskas      | 1983-08-23 | Treneris                |
------------------------------------------------------------------------------------------------------------------
```

```
Console(2):
```



```
Nariai dalyvave visose stovyklose:
----------------------------------------------------------------------------------------------------------
Nariu nera
----------------------------------------------------------------------------------------------------------

Visi dalyvave puolejai:
----------------------------------------
Nariu nera
----------------------------------------

Vyr. treneriai:
--------------------------------
Nariu nera
--------------------------------

1-os stovyklos dalyviai:
----------------------------------------------------------------------------------------------------------
| Sigitas         | Valeika         | 1980-11-11 | Treneris         |
| Andrius         | Jankunas        | 1994-07-21 |     1,86 | Izaidejas | Klaipedos Neptunas    | False | False |
| Dovile          | Petrylaite      | 1995-07-18 | Masazuotoja      |
| Alvydas         | Jogaila         | 1995-11-27 |     1,89 | Gynejas   | Alytaus Alita         | False | True  |
| Tomas           | Andriulis       | 1996-05-03 |     1,88 | Izaidejas | Taurages Taurage      | False | False |
----------------------------------------------------------------------------------------------------------

2-os stovyklos dalyviai:
----------------------------------------------------------------------------------------------------------
| Algis           | Roleika         | 1982-04-05 | Treneris         |
| Petras          | Adomavicius     | 1993-01-15 |     1,96 | Gynejas   | Utenos Juventus       | False | True  |
| Jonas           | Kazakevicius    | 1993-03-26 |     1,92 | Gynejas   | KK Lusis              | False | False |
| Daiva           | Giraite         | 1993-12-17 | Masazuotoja      |
| Alvydas         | Jogaila         | 1995-11-27 |     1,89 | Gynejas   | Alytaus Alita         | False | True  |
----------------------------------------------------------------------------------------------------------

3-os stovyklos dalyviai:
----------------------------------------------------------------------------------------------------------
| Sigitas         | Valeika         | 1980-11-11 | Treneris         |
| Ramunas         | Nerkauskas      | 1983-08-23 | Treneris         |
| Petras          | Adomavicius     | 1993-01-15 |     1,96 | Gynejas   | Utenos Juventus       | False | True  |
| Jonas           | Kazakevicius    | 1993-03-26 |     1,92 | Gynejas   | KK Lusis              | False | False |
| Andrius         | Jankunas        | 1994-07-21 |     1,86 | Izaidejas | Klaipedos Neptunas    | False | False |
| Tomas           | Andriulis       | 1996-05-03 |     1,88 | Puolejas  | Taurages Taurage      | False | False |
----------------------------------------------------------------------------------------------------------
```

```
Rinktine.csv(2):
```



Žaidėjai pakviesti į rinktinę:	
Narių nėra	

5.4. Dėstytojo pastabos