PROGRAMOVÁNÍ

1	**SPŠ Chomutov**	**Beránek**
16.12.2022	**Diář**	**V4**

Zadání:

Vytvořte program, který bude spravovat diář a využijte při tom vlastností OOP. Události v diáři by měla být objektem, tedy popsán a definován třídou. Událost by měla mít tyto vlastnosti:

- Popis události (řetězec)
- Začátek (časové razítko)
- Délka trvání události a/nebo konec události
- Priorita (bude zobrazeno jako barevná značka nebo proužek
- Příznak, že událost je celodenní
- Příznak, že událost je smazaná

Připravte přetížený konstruktor funkce tak, aby existovaly tři možné varianty vytvoření nové události

- Bez parametrů - a tedy všechny vlastnosti události budou defaultní
- S dvěma parametry - Popis události + časové razítko začátku
- S pěti parametry - Popis, Začátek, Trvání, Priorita a Bool „celodenní“

Teorie:

Objektově orientované programování se zaměřuje více na to, jak program vidí člověk spíše než stroj. Využívá práce s objekty, které mají vlastní atributy a metody. Protože je všechno uspořádané pod objekty je kód přehlednější, a hlavně čitelnější, na první pohled je mnohem jasnější co má program dělat. Objekty se vytváří ze tříd. Ty slouží jako jakési šablony pro vytváření objektů. Z jedné třídy můžeme vytvořit několik objektů, které budou sdílet stejné atributy a metody, ale budou mít jiná data.

Popis programu:

Při stisknutí tlačítka „Nová Událost“ se vytvoří nový objekt *udalost* ze třídy *Event*, a zavolá se funkce *udalost.priradit* a jako argument je použit list *dny*. Pokud funkce vrátí false, je vytvořen nový objekt *den* podle třídy *Day* s argumentem *udalost.Start.Date*, objekt je přidán do listu *dny* a *udalost* je přidána do listu *den.Events*. Nakonec je *den* přiřazen do listboxu *list_days*.

Funkce *udalost.priradit* kontroluje všechny objekty v listu *list* jestli se *Start.Date* proměnná rovná proměnné *list[i].Date*. Pokud najde takový objekt, přiřadí do jeho listu *Events* tuto *udalost* a vrátí true. Pokud nenajde, vrátí false.

Při kliknutí na tlačítka „Nová Událost +“ nebo „Nová Událost Pro“ je průběh stejný, akorát se používá jiné přetížení konstruktoru. Data zadaná uživatelem jsou uložena ve formu v public proměnných, a ty jsou převzata a použita jako argumenty pro konstruktor.

Při kliknutí v listboxu *list_days* se zkontroluje index vybraného předmětu. Pokud je index roven -1, je funkce ukončena. Pokud ne, listbox *list_events* se vyčistí a následně je z listu *dny* vybrán předmět podle indexu vybraného předmětu v listboxu *list_days* a je zavolána jeho funkce *vypsat* s argumentem *list_events*.

Funkce *Day.vypsat* vypíše obsah listu *Day.Events* do listboxu, který funkce obdržela jako argument.

Při kliknutí v listboxu *list_events* se zkontroluje index vybraného předmětu. Pokud je index roven -1, je funkce ukončena. Pokud ne, v listu *dny* se vybere objekt podle indexu vybraného předmětu v listboxu *list_days*, a z jeho listu *Events* se vybere objekt podle indexu vybraného předmětu v listboxu *list_events* a zavolá se funkce *vypsat* s argumentem *txb_event*.

Funkce *Event.vypsat* je součástí třídy *Event*. Její funkce spočívá ve využití svých interních proměnných, jejich složení do jednoho stringu s příslušným odřádkováním, a následném vypsaní do textboxu, který funkce obdržela jako argument.

Při dvojkliku na listbox *list_events* se zkontroluje index vybraného předmětu. Pokud je index roven -1, je funkce ukončena. Pokud ne, je vybrána událost z listu *Days.Events* pomocí přislušných indexů v listboxech, a tato událost je použita jako argument pro konstruktor *Form_newEventPro*. Pokud dialog vrátí výsledek „OK“, jsou data uložená v proměnných formu použita pro přepsaní dat vybrané události. Nakonec je v listu *list_events* název aktuální události přepsán novým názvem a zavolána funkce *Event.vypsat* z aktuální události.

Při zmáčknutí tlačítka „Odstranit“ se vybere událost z listu *Days.Events* pomocí příslušných indexů v listboxech, je označena jako deleted a je přidána do listu *odstranene*, který je součást formu *Form_bin*. Následně je událost odstraněna z listu *Days.Events* a z listboxu *list_events*. Dále je vyčištěn textbox *txb_event*. Nakonec je zkontrolován počet předmětů v listu *Days.Events* aktuálně vybraného dne. Pokud je počet roven nule, je den odstraněn z listu *dny* a z listboxu *list_days*.

Při stisknutí tlačítka „Koš“ se otevře *Form_bin*. Ten při načítání vyčistí list *obnovene*, vyčistí obsah textboxu *txb_event* a vypíše všechny události v listu *odstranene* do listboxu *list_udalosti*. Při stisknutí tlačítka „Obnovit“ se událost přesune do listu *obnovene* a odstraní se z listu *odstranene* a z listboxu *list_udalosti*. Při stisknutí tlačítka „Vysypat Koš“ se vyčistí list *odstranene* a listbox *list_udalosti*. Při zavření formu *Form_bin* se zkontroluje počet předmětů v listu *obnovene*. Pokud je větší jak nula, je všem událostem v listu odstraněn příznak odstranění a následně jsou přiřazeny do příslušného dne pomocí funkce *Event.priradit*.

Seznam proměnných a funkcí:

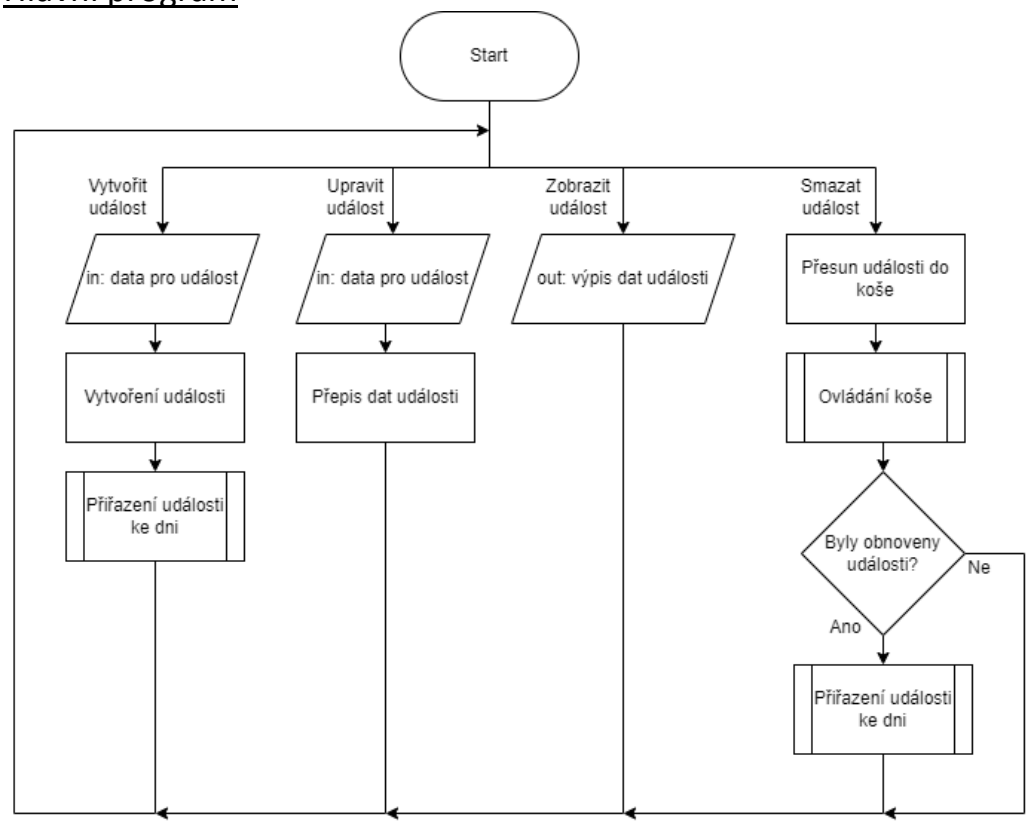
Proměnná	Účel
Class Day	
public DateTime Date	Datum daného dne
public List<Event> Events	Seznam událostí začínajících tento den
Class Event	
public string Desc	Popis/název události
public DateTime Start	Začátek události
public DateTime End	Konec události

public int Priority	Priorita události
public bool AllDay	Příznak, jestli je událost celodenní
public bool Deleted	Příznak, jestli je událost smazána
Class Form_main	
public List<Day> dny	Seznam vytvořených dní
Class Form_newEventPlus	
public string desc	Popis vytvářené události
public DateTime start	Začátek vytvářené události
Class Form_newEventPro	
public string desc	Popis vytvářené události
public DateTime start	Začátek vytvářené události
public DateTime end	Konec vytvářené události
public bool allday	Příznak, jestli je vytvářená událost celodenní
public int priority	Příznak, jestli je vytvářená událost smazána
Class Form_bin	
public List<Event> odstranene	Seznam odstraněných událostí
public List<Event> obnovene	Seznam obnovených událostí

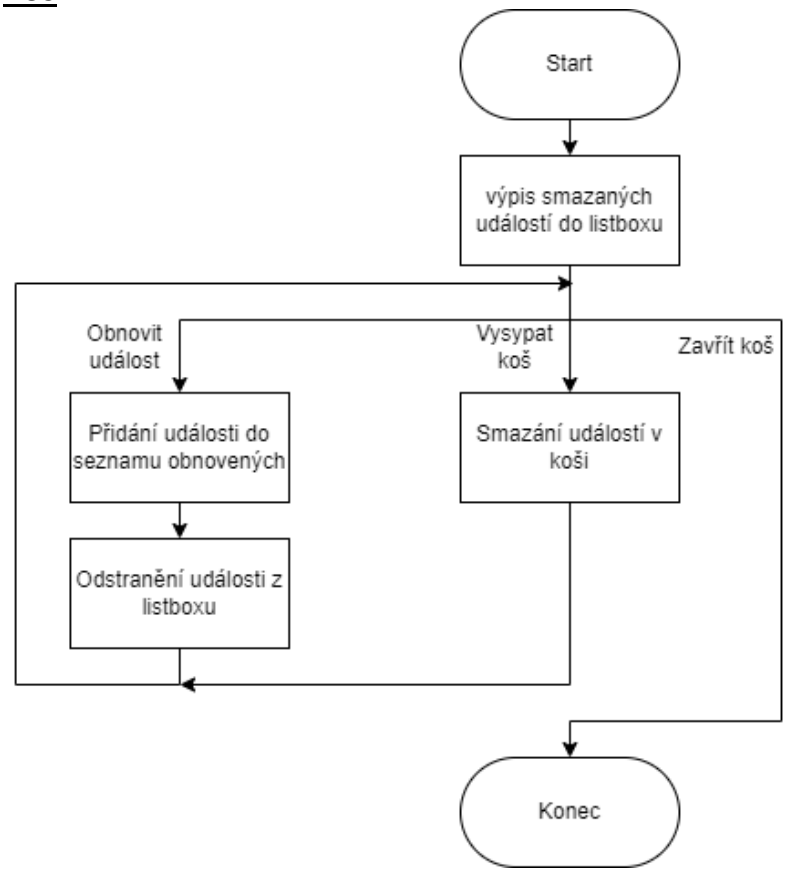
Funkce	Účel
Class Day	
public void vypsat(ListBox lsb)	Vypíše seznam událostí do listboxu
Class Event	
public void vypsat(TextBox txb)	Vypíše podrobnosti o události do textboxu
public bool priradit(List<Day> list)	Přiřadí událost ke dni z listu podle začátku události Pokud v listu nenajde odpovídající den, vrátí false

Vývojový diagram:

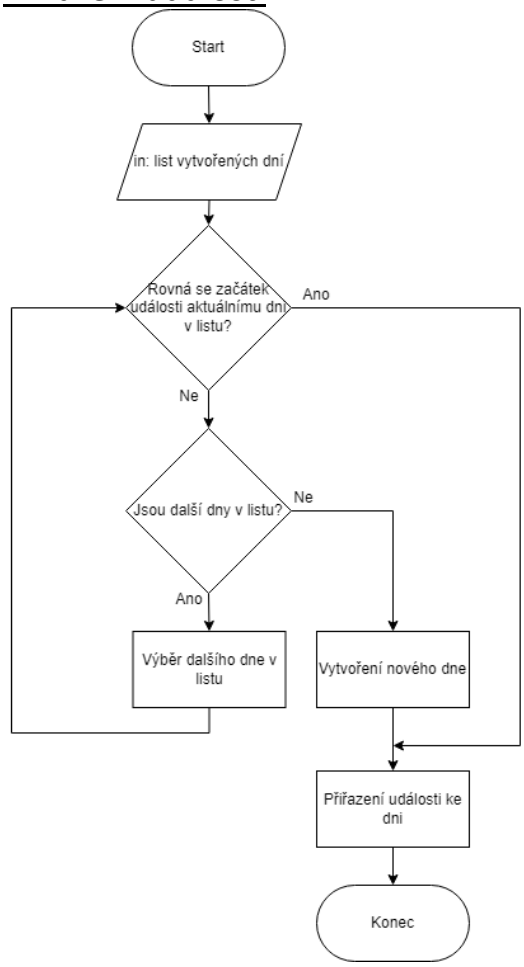
Hlavní program



Koš



Přiřazení události



Zdrojový kód:

Form_main.cs

```
1. public partial class Form_main : Form
2.     {
3.         public List<Day> dny = new List<Day>();
4.         Form_bin bin = new Form_bin();
5.
6.         public Form_main()
7.         {
8.             InitializeComponent();
9.         }
10.        private void novaUdalost_click(object sender, EventArgs e)
11.        {
12.            Event udalost = new Event();
13.            if(!udalost.priradit(dny)) // Pokud funkce vrátí false, událost nebyla přiřazena a musí se
vytvořit nový den
14.            {
15.                Day den = new Day(udalost.Start.Date);
16.                dny.Add(den);
17.                den.Events.Add(udalost);
18.                list_days.Items.Add(den.Date);
19.            }
20.            if (list_days.SelectedIndex != -1) dny[list_days.SelectedIndex].vypsat(list_events);
21.        }
22.
23.        private void list_events_click(object sender, EventArgs e)
24.        {
25.            if (list_events.SelectedIndex == -1) return;
26.            dny[list_days.SelectedIndex].Events[list_events.SelectedIndex].vypsat(txb_event);
27.        }
28.
29.        private void btn_newEventPlus_Click(object sender, EventArgs e)
30.        {
31.            Form_newEventPlus form = new Form_newEventPlus();
32.            if(form.ShowDialog() == DialogResult.OK)
33.            {
34.                Event udalost = new Event(form.desc, form.start);
35.                if(!udalost.priradit(dny)) // Pokud funkce vrátí false, událost nebyla přiřazena a musí
se vytvořit nový den
36.                {
37.                    Day den = new Day(udalost.Start.Date);
38.                    dny.Add(den);
39.                    den.Events.Add(udalost);
40.                    list_days.Items.Add(den.Date);
41.                }
42.                if (list_days.SelectedIndex != -1) dny[list_days.SelectedIndex].vypsat(list_events);
43.            }
44.        }
45.
46.        private void btn_newEventPro_Click(object sender, EventArgs e)
47.        {
48.            Form_newEventPro form = new Form_newEventPro();
49.            if(form.ShowDialog() == DialogResult.OK)
50.            {
51.                Event udalost = new Event(form.desc, form.start, form.end, form.priority, form.allday);
52.                if(!udalost.priradit(dny)) // Pokud funkce vrátí false, událost nebyla přiřazena a musí
se vytvořit nový den
53.                {
54.                    Day den = new Day(udalost.Start.Date);
55.                    dny.Add(den);
56.                    den.Events.Add(udalost);
57.                    list_days.Items.Add(den.Date);
58.                }
59.                if (list_days.SelectedIndex != -1) dny[list_days.SelectedIndex].vypsat(list_events);
60.            }
61.        }
62.
63.        private void list_events_DoubleClick(object sender, EventArgs e)
64.        {
65.            if (list_events.SelectedIndex == -1) return;
66.            Event udalost = dny[list_days.SelectedIndex].Events[list_events.SelectedIndex];
67.            Form_newEventPro form = new Form_newEventPro(udalost);
68.            if(form.ShowDialog() == DialogResult.OK)
```

```
            {
                udalost.Desc = form.desc;
                udalost.Start = form.start;
                udalost.End = form.end;
                udalost.AllDay = form.allday;
                udalost.Priority = form.priority;
                list_events.Items[dny[list_days.SelectedIndex].Events.IndexOf(udalost)] = form.desc;
                udalost.vypsat(txb_event);
            }
        }

        private void btn_delete_Click(object sender, EventArgs e)
        {
            if (list_events.SelectedIndex == -1 || list_days.SelectedIndex == -1) return;
            dny[list_days.SelectedIndex].Events[list_events.SelectedIndex].Deleted = true;
            bin.odstranene.Add(dny[list_days.SelectedIndex].Events[list_events.SelectedIndex]);
            dny[list_days.SelectedIndex].Events.RemoveAt(list_events.SelectedIndex);
            list_events.Items.RemoveAt(list_events.SelectedIndex);
            txb_event.Text = "";
            if(dny[list_days.SelectedIndex].Events.Count == 0)
            {
                dny.RemoveAt(list_days.SelectedIndex);
                list_days.Items.RemoveAt(list_days.SelectedIndex);
            }
        }

        private void btn_trash_Click(object sender, EventArgs e)
        {
            bin.ShowDialog();
            if (bin.obnovene.Count != 0)
            {
                for (int i = 0; i < bin.obnovene.Count; i++)
                {
                    bin.obnovene[i].Deleted = false;
                    if (!bin.obnovene[i].priradit(dny))
                    {
                        Day den = new Day(bin.obnovene[i].Start.Date);
                        dny.Add(den);
                        den.Events.Add(bin.obnovene[i]);
                        list_days.Items.Add(den.Date);
                    }
                }
                if (list_days.SelectedIndex != -1) dny[list_days.SelectedIndex].vypsat(list_events);
            }
        }

        private void list_days_Click(object sender, EventArgs e)
        {
            if (list_days.SelectedIndex == -1) return;
            list_events.Items.Clear();
            dny[list_days.SelectedIndex].vypsat(list_events);
        }
    }
```

Form_bin.cs

```
1. public partial class Form_bin : Form
    {

        public List<Event> odstranene = new List<Event>();
        public List<Event> obnovene = new List<Event>();

        public Form_bin()
        {
            InitializeComponent();
        }

        private void Form_bin_Load(object sender, EventArgs e)
        {
            obnovene.Clear();
            txb_event.Text = "";
            if (odstranene.Count != 0 && odstranene.Count != list_udalosti.Items.Count)
            {
                for (int i = 0; i < odstranene.Count; i++)
                {
                    list_udalosti.Items.Add(odstranene[i].Desc);
```

```
                }
            }
        }

        private void list_udalosti_Click(object sender, EventArgs e)
        {
            if (list_udalosti.SelectedIndex == -1) return;
            odstranene[list_udalosti.SelectedIndex].vypsat(txb_event);
        }

        private void btn_restore_Click(object sender, EventArgs e)
        {
            if (list_udalosti.SelectedIndex == -1) return;
            obnovene.Add(odstranene[list_udalosti.SelectedIndex]);
            odstranene.RemoveAt(list_udalosti.SelectedIndex);
            list_udalosti.Items.RemoveAt(list_udalosti.SelectedIndex);
        }

        private void btn_removeall_Click(object sender, EventArgs e)
        {
            odstranene.Clear();
            list_udalosti.Items.Clear();
            txb_event.Text = "";
        }
    }
```

Event.cs

```
public class Event
    {
        public string Desc;
        public DateTime Start;
        public DateTime End;
        public int Priority;
        public bool AllDay;
        public bool Deleted;

        public Event()
        {
            Desc = "Nová událost";
            Start = DateTime.Now;
            End = Start + TimeSpan.FromHours(1);
            Priority = 1;
            AllDay = false;
            Deleted = false;
        }

        public Event(string User_desc, DateTime User_start)
        {
            Desc = User_desc;
            Start = User_start;
            End = Start + TimeSpan.FromHours(1);
            Priority = 1;
            AllDay = false;
            Deleted = false;
        }

        public Event(string User_desc, DateTime User_start, DateTime User_end, int User_priority, bool
User_allday)
        {
            Desc = User_desc;
            Start = User_start;
            End = User_end;
            Priority = User_priority;
            AllDay = User_allday;
            Deleted = false;
        }
        /// <summary>
        /// Vypíše vlastnosti do textboxu
        /// </summary>
        /// <param name="txb">
        /// Textbox, do kterého se vlastnosti vypíší
        /// </param>
        public void vypsat(TextBox txb)
        {
            List<string> list = new List<string>
```

```
            {
                "nízká", "střední", "vysoká"
            };
            string text;
            string priorita = list[Priority];
            string celodenni = (AllDay) ? "ano" : "ne";
            text = $"{Desc}\r\nZačátek: {Start}\r\nKonec: {End}\r\nPriorita: {priorita}\r\nCelodenní:
{celodenni}";
            txb.Text = text;
        }

        /// <summary>
        /// Pokusí se přiřadit událost k odpovídajícímu dni
        /// </summary>
        /// <param name="list">
        /// list dní, ve kterých se hledá odpovídající den
        /// </param>
        /// <returns>
        /// true: Událost byla přiřazena ke dni
        /// false: Den nebyl nalezen, událost nebyla přiřazena
        /// </returns>
        public bool priradit(List<Day> list)
        {
            for (int i = 0; i < list.Count; i++)
            {
                if (Start.Date == list[i].Date)
                {
                    list[i].Events.Add(this);
                    return true;
                }
            }
            return false;
        }
    }
```

Day.cs

```
public class Day
    {
        public DateTime Date;
        public List<Event> Events = new List<Event>();

        public Day(DateTime date)
        {
            Date = date;
        }
        /// <summary>
        /// Vypíše všechny přiřazené události
        /// </summary>
        /// <param name="lsb">
        /// Listbox, do kterého se události vypíší
        /// </param>
        public void vypsat(ListBox lsb)
        {
            lsb.Items.Clear();
            for (int i = 0; i < Events.Count; i++)
            {
                lsb.Items.Add(Events[i].Desc);
            }

        }
    }
```

Závěr:

Program funguje, jak má, ale je místy nepřehledný a ne moc efektivní.