UNIVERZITA PARDUBICE

FAKULTA ELEKTROTECHNIKY A INFORMATIKY

BAKALÁŘSKÁ PRÁCE

2017 Daniel Jílek

Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Hra TANGRAM SQUARE pro platformu Windows

Daniel Jílek

Bakalářská práce

2017

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2016/2017**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Daniel Jílek**
Osobní číslo:	**I14108**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Hra TANGRAM SQUARE pro platformu Windows**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem bakalářské práce je vytvořit počítačovou hru na principu puzzle "tangram" a vytvořit obrazec (čtverec) z dílčích polygonů.

V teoretické části bude představen problém puzzle "tangram" a principy, jak rozpoznat, že sestavený tvar odpovídá zadanému požadovanému. Hráč může jednotlivé části libovolně posouvat, otáčet a spojovat, algoritmus by měl být schopen rozpoznat libovolnou kombinaci dílků, která korektně představuje zadaný tvar (použity všechny dílky, nepřekrývají se, bez prázdných míst uvnitř obrazce).

V praktické části bude hra realizována a budou pro ni připraveny minimálně 3 různé sady dílků k otestování. Program bude vytvořen v jazyku Java nebo C#.

Rozsah grafických prací:

Rozsah pracovní zprávy: **30-40 stran**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

RICHARDSON, W. Clay. Professional Java, JDK. 5th ed. Indianapolis, IN: Wrox, c2005, xxxi, 712 p. ISBN 07-645-7486-8.

NAGEL CH. et al. C# 2008. Programujeme profesionálně. Brno, 2009. ISBN 978-80-251-2407-7.

Tangram. Wikipedia: The Free Encyclopedia [online]. Wikimedia Foundation [cit. 2016-10-21]. Dostupné z: https://en.wikipedia.org/wiki/Tangram

Vedoucí bakalářské práce: **Ing. Roman Diviš**
Katedra softwarových technologií

Datum zadání bakalářské práce: **31. října 2016**

Termín odevzdání bakalářské práce: **12. května 2017**

Univerzita Pardubice Česká republika

L.S.

Ing. Zdeněk Němec, Ph.D.
děkan

Mgr. Josef Horálek, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2017

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 11. 05. 2017

Daniel Jílek

PODĚKOVÁNÍ

Děkuji Ing. Romanu Divišovi za odborné vedení, trpělivost a ochotu, kterou mi věnoval při zpracování této bakalářské práce.

ANOTACE

Teoretická část práce se zabývá detailním popisem mechanické logické hry Tangram. Věnuje se historii, základní charakteristice a pravidlům hry. Rovněž je představena problematika spojena se zvláštnostmi Tangramu. Poté je představena aplikace Tangram, která byla pro potřeby praktické části práce implementována v programovacím jazyce Java. Aplikace je zdokumentována prostřednictvím uživatelské a technické dokumentace.

KLÍČOVÁ SLOVA

Tangram, Java, aplikace

TITLE

The Tangram square game for Windows platform

ANNOTATION

In the theoretical part of the thesis detailed description of the mechanical logic game called Tangram can be found. There is described history, basic characteristics and the rules of the game. There are also introduced issues associated with Tangram. After that, there is a description of implementation of the application. Tangram is implemented in programming language Java. Application is described by user and technical documentation.

KEYWORDS

Tangram, Java, application

OBSAH

SEZNAM ILUSTRACÍ A TABULEK

SEZNAM ZKRATEK A ZNAČEK

BSD	Berkeley Software Distribution
CSS	Cascading Style Sheets
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
LTM	Levé tlačítko myši
PTM	Pravé tlačítko myši
WYSIWYG	What you see is what you get
XML	Extensible Markup Language

ÚVOD

Tématem bakalářské práce je mechanická logická hra *Tangram* pocházející z Číny. Její původ sice sahá až do doby 4000 let př. n. l., rozmach této hry po celém světě ale nastal až na konci devatenáctého století. *Tangram* se skládá ze sedmi geometrických útvarů, mezi něž patří dva velké trojúhelníky, jeden střední trojúhelník, dva malé trojúhelníky, čtverec a rovnoběžník. Různými kombinacemi spojení těchto dílků mohou vznikat všemožné obrazce. Pravidla hry jsou velice jednoduchá. Žádný z dílků se nesmí vzájemně překrývat a všechny dílky musí tvořit předem stanovený obrazec. Dílky *Tangramu* je během hry možné otáčet i převracet.

Cílem práce je blíže představit tuto logickou hru, její historii, základní charakteristiku a její zvláštnosti. Většina zvláštností této hry je spojena s optickými iluzemi. V praktické části je pak implementována aplikace *Tangram* pro platformu Windows. Tato aplikace by měla umět rozpoznat jakoukoliv kombinaci dílku správně umístěnou uvnitř podkladu bez jakéhokoliv porušení pravidel hry. Uživateli dává na výběr tvar, který chce skládat. Následně hráč dílky umisťuje dovnitř podkladu.

V první části se práce věnuje výkladu historie *Tangramu*. Jsou popsána pravidla hry a způsoby, jak *Tangram* hrát. V další kapitole se práce věnuje paradoxům *Tangramu* a konvexním *Tangramům*. Poslední kapitola první části představuje hry, které jsou *Tangramu* podobné. V druhé části práce je detailně popsána aplikace *Tangram* pro platformu Windows prostřednictvím uživatelské a technické dokumentace. Uživatelská dokumentace obsahuje popis funkčností. Technická dokumentace se věnuje zdrojovým kódům, nebo algoritmům, které aplikace využívá.

Protože se jedná o starou logickou hru, existuje malé množství nové literatury, která by se tímto tématem zabývala. Literatura, z které bylo pro účely této práce čerpáno, je proto většinou starší. To však nic nemění na aktuálnosti informací získaných z těchto knih, protože má hra stále stejná pravidla a v průběhu let se vůbec nezměnila.

1 TANGRAM

Tangram je mechanická logická hra pocházející z Číny, jejíž vznik sahá až do doby 4000 let př. n. l. Skládá se ze sedmi geometrických útvarů s rovnými hranami, které se nazývají Tany. Hlavní kouzlo hry spočívá v nepřeberném množství kombinací, do jakých je možné *Tangram* složit (Loyd, 2007, s.1). V následujících podkapitolách je blíže nastíněna historie tohoto hlavolamu, pravidla hry a způsoby, jakými *Tangram* hrát. Dále jsou popsány a vysvětleny paradoxy *Tangramu*. Předposlední podkapitola se věnuje konvexním tvarům, které je možné složit za pomoci dílků *Tangramu*. Nakonec jsou zmíněny skládačky, které jsou *Tangramu* velice podobné.

1.1 Historie

Jedná se o velmi starou logickou hru pocházející z doby 4000 let př. n. l. Přesto se první novodobé zmínky o *Tangramu* vyskytují až na začátku devatenáctého století v Číně. V té době se začaly objevovat knihy věnující se tomuto tématu. Krátce na to se hra rozšířila do ostatních koutů světa, nejprve do Evropy, následně do Ameriky (Read, 1965, s. 2-3).

1.1.1 Název

První zmínka slova *Tangram* ve Websterově slovníku pochází z roku 1864. Slovo bylo pravděpodobně vymyšleno Britským nebo Americkým výrobcem hraček, neexistují však spolehlivé důkazy (Loyd, 2007, s. VII). Osoba, která se snažila hru pojmenovat, zřejmě vycházela ze slova „t'ang", což v kantonském dialektu znamená „čínský". V Čínské literatuře, historii ani tradicích neexistuje zmínka o člověku nebo bohu jménem Tan. Doslovně přeložený název skládačky zní „geniální skládačka ze sedmi dílků" (Dudeney, 2007, s. 37).

1.1.2 Legenda

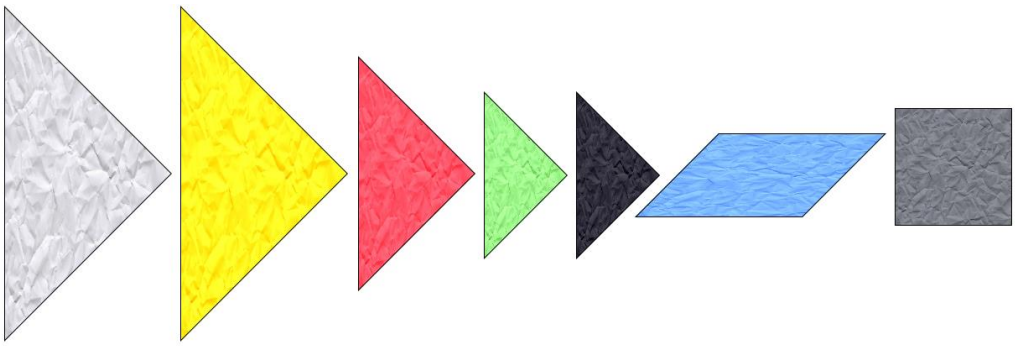
O původu a vzniku *Tangramu* existuje velké množství legend. Jedna z nich vypráví o řemeslníkovi jménem Tan, který pro císaře vyrobil zdobenou dlaždici. Když, šel s darem po chodbě paláce, upadl a upustil dlaždici na zem. Ta se rozpadla na sedm kousků. Tan se okamžitě pokoušel dílky spojit zpět dohromady, ale jak se snažil, vznikaly mu stále jiné geometrické tvary. Služebníci, kteří byli císařem poslání zjistit co se děje, byli ohromeni všemožnými kombinacemi a předali rozbité dílky císařovi jako hlavolam pro jeho potěšení (*Legendy o vzniku Tangramu*, 2012).

1.2 Základní charakteristika hry

V následujících kapitolách je popsán vzhled *Tangramu*, jeho rozměry, pravidla hry a způsoby, kterými je možné hru hrát.

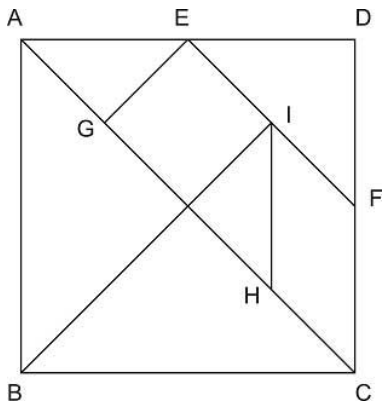
1.2.1 Složení Tangramu

Tangram se skládá ze sedmi dílků se specifickými rozměry. Konkrétně hra obsahuje dva velké trojúhelníky, jeden střední trojúhelník, dva malé trojúhelníky, rovnoběžník a čtverec. (Read, 1965, s. 1). Dílky jsou znázorněny na obrázku 1.



Obrázek 1 – Dílky Tangramu

Nejjednodušším způsobem pro získání dílků je rozřezání čtverce podle daných pravidel. Rohy čtverce jsou označeny body A, B, C, D. V prvním kroku je označen bod E, ležící přesně uprostřed mezi body A a D. Následně je označen bod F ležící mezi body D a C. Spojením bodů E a F je získán střední trojúhelník. V dalším kroku je spojen bod A a C. Vytvořením bodu I ležícím uprostřed mezi body E a F a spojením bodu I s bodem B jsou získány dva velké trojúhelníky. Na přímce mezi body A a C jsou označeny další dva body G a H ležící na středech odvěsen velkých trojúhelníků. Spojením bodu G a E je získán malý trojúhelník a čtverec. Spojením bodu H a I je získán malý trojúhelník a rovnoběžník (*Rozměry tanů*, 2012). Všechny body jsou znázorněny na obrázku 2.



Obrázek 2 – Půdorys (*Rozměry tanů*, 2012)

Po rozřezání čtverce na dílky musí být dodržena následující pravidla (*Rozměry tanů*, 2012):

- body A, B, C, D tvoří čtverec;
- |AE| = |DE| = |DF| = |CF| = |HI|;
- |EI| = |FI|.

Všechny vzniklé trojúhelníky jsou pravoúhlé a rovnoramenné. Vnitřní úhly všech dílků jsou násobkem 45 °, dílky mají tedy vnitřní úhly o velikostech 45 °, 90 ° a 135 °. Střední trojúhelník má spolu s čtvercem a rovnoběžníkem totožný obsah. Plocha malého trojúhelníku je polovinou obsahu středního trojúhelníku a plocha středního trojúhelníku je polovinou obsahu velkého trojúhelníku (*Rozměry tanů*, 2012).

1.2.2 Pravidla

Při sestavování dílků je nutné dodržet tato pravidla (Zapletal, 1983, s. 26):

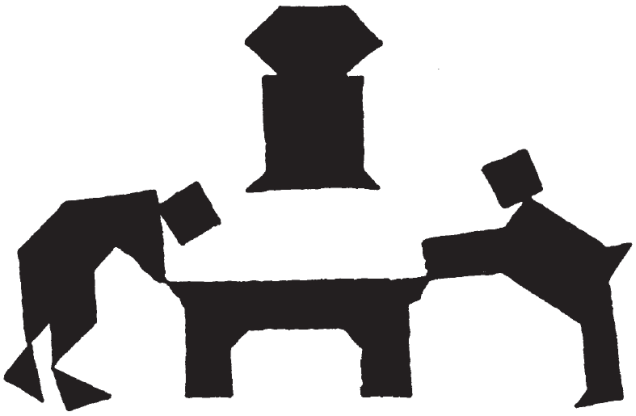
- Obrys složených dílků musí odpovídat předloze.
- Pro složení *Tangramu* je povinné použít všechny dílky. Žádný dílek nesmí přebývat.
- Dílky musí ležet na podložce a nesmí se vzájemně překrývat.
- Všechny dílky musí být uvnitř předlohy.
- S dílky je možné otáčet a je možné je převracet.

1.2.3 Způsob hry

Tangram je možné hrát několika způsoby. Mezi ně patří (Gardner, 1988, s. 29):

- Hledání jedné, nebo více možností sestavení *Tangramu* podle předem daného tvaru.
- Vytváření nových rozpoznatelných tvarů.
- Řešení a objevování řady geometrických problémů či iluzí.

Další možností, jak hrát *Tangram*, je skládání mnohem komplexnějších obrazců za pomoci více sad dílků. *Tangram* složený ze dvou čí více sad je mnohem složitější sestavit a dává možnost tvořit mnohem ambicióznější tvary, či obrazy. Na obrázku 3 hrají dvě osoby kulečník a v pozadí mají hodiny. Obraz je složený ze čtyř sad dílků *Tangramu* (Dudeney, 2007, s. 38-39).

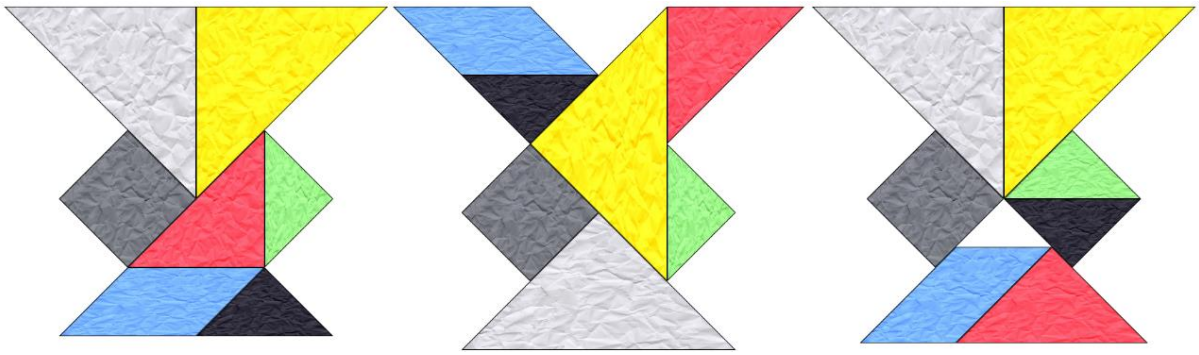


Obrázek 3 – Dva hráči hrající kulečník s hodinami v pozadí (Dudeney, 2007, s. 39)

1.3 Paradoxy Tangramu

Tangram může se svými dílky často poskytnout velice kuriózní situace, které jsou v této kapitole představeny. Převážně se jedná o optické iluze, dosažené správným uspořádáním dílků.

Na obrázku 4 jsou vyobrazeny tři složené tvary ve tvaru vázy, které jsou na první pohled velmi podobné. Přestože jsou všechny vázy stejně velké, mají stejný tvar a na všechny bylo použito všech sedm dílů skládačky, na druhé a třetí váze část chybí. Třetí váza má dokonce chybějící část menší než váza druhá (Loyd, 2007, s. 25).

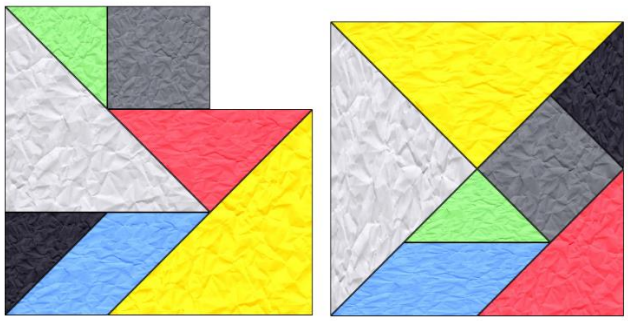


Obrázek 4 – Vázy

Ve skutečnosti se jedná pouze o optickou iluzi. Při bližším zkoumání je možné zjistit, že druhá a třetí váza jsou vyšší než první a tím kompenzují chybějící prostor.

Podle pravidel *Tangramu* je nutné využít vždy všech sedm dílků. Někdy však stačí ke složení požadovaného tvaru dílků pouze šest, nebo naopak jeden dílek může chybět. Na obrázku 5 jsou vyobrazeny dva pokusy o sestavení čtverce. Na prvním je možné pozorovat, že by k jeho

dokončení byl potřeba osmý dílek ve tvaru čtverce. Druhý čtverec je úspěšně složen za pomoci sedmi dílků (Loyd, 2007, s. 25-26).



Obrázek 5 – Čtverce

1.3.1 Matematické problémy

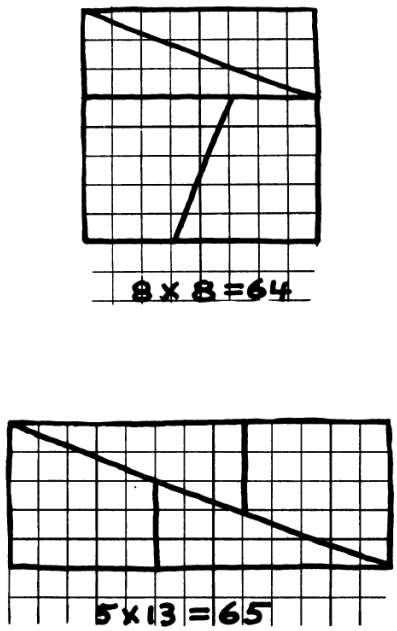
Pokud je stanovena délka čtverce složeného z dílků na 34 cm, výsledný obsah činí 1156 cm^2. Dále je možné zjistit že jsou dílky *Tangramu* definovány pouze čtyřmi rozměry. Při zachování délky strany výsledného čtverce na 34 cm jsou délky stran dílků 34 cm, 24 cm, 17 cm a 12 cm (Loyd, 2007, s. 28).

Obrázek 6 – Rozměry stran

Velký trojúhelník má obsah 288 cm^2 a malý trojúhelník 72 cm^2. Střední trojúhelník a rovnoběžník mají i přes odlišný obvod stejný obsah, který činí 144,5 cm^2. Zbývá čtverec s obsahem 144 cm^2. Při spočtení obsahu složeného *Tangramu* o délce strany 34 cm, byl obsah 1156 cm^2. Při sečtení obsahů jednotlivých dílků je však získán obsah o velikosti 1153 cm^2 (Loyd, 2007, s. 28).

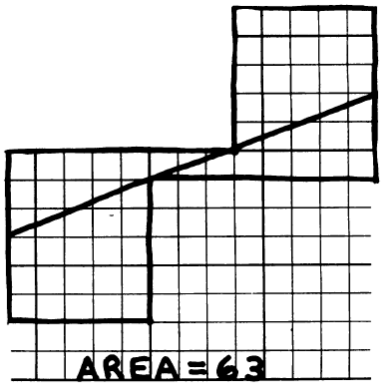
1.3.2 Hooperův paradox

Problém z předchozí kapitoly také popisuje čtvercová varianta Hooperova paradoxu. Základem je čtverec o rozměrech 8 × 8 dílků. Čtverec je rozřezán na dva trojúhelníky a dva lichoběžníky. Složený čtverec obsahuje 64 dílků. Pokud je čtverec přeskládán tak, aby tvořil obdélník o rozměrech 5 × 13 dílků, jejich počet se ze 64 zvětší o jeden (Gardner, 1956, s. 132-133).



Obrázek 7 – Hooperův paradox I (Gardner, 1956, s. 133)

Dílky mohou být dokonce poskládány tak, aby byl výsledný počet dílků o jeden menší než u původního čtverce, viz obrázek 8.



Obrázek 8 – Hooperův paradox II (Gardner, 1956, s. 134)

Aby byl tento paradox zjevný, je nutno při rozdělování čtverce dodržet pravidla Fibonacciho posloupnosti. Ve Fibonacciho posloupnosti je každé následující číslo součtem dvou

čísel předchozích. Začátek posloupnosti vypadá následovně: 0, 1, 1, 2, 3, 5, 8, 13, 21, … (Gardner, 1956, s. 134-135).

Pro demonstraci Hooperova paradoxu byl zvolen čtverec rozdělený na dílky o rozměrech 8 × 8 dílků. Osmička je ve Fibonacciho posloupnosti obklopena čísly 5 a 13. Tato čísla určují výsledné rozměry obdélníku, který je složen po přeskládání dílků čtverce. Dvě čísla v posloupnosti před osmičkou jsou čísla 3 a 5. Tyto hodnoty určují, v jaké části je potřeba čtverec rozpůlit. Jedna polovina má výšku tři dílky a druhá má výšku pět dílků. Stejná pravidla platí pro všechny rozměry vyskytující se ve Fibonacciho posloupnosti. Pokud je například zvolen rozměr čtverce 13 × 13, výsledný obdélník má rozměry 21 × 8 a čtverec je potřeba rozpůlit mezi pátým a šestým řádkem (Gardner, 1956, s. 135-136).

Jsou vybrána tři po sobě jdoucí čísla z Fibonacciho posloupnosti označena písmeny A, B a C. Písmenem X je označena hodnota představující ztrátu nebo zisk obsahu. Z těchto údajů je možné odvodit vzorec (Gardner, 1956, s. 137):

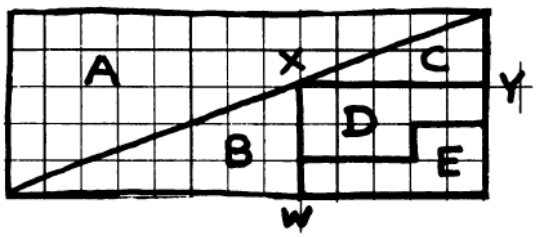
$$A + B = C$$

$$B^2 = AC \pm X$$

1.3.3 Curryho paradox

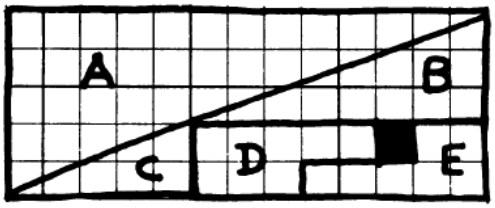
Všechny tyto případy mizejícího dílku jsou pouze optickým klamem. Dílky ve skutečnosti obsahují stále stejně velkou plochu, jejich obsah se nemění. Problém dobře popisuje Curryho paradox.

Rozřezáním obdélníku o rozměrech 5 × 13 dílků jsou získány tři trojúhelníky A, B a C a dva polygony D a E, viz obrázek 9.



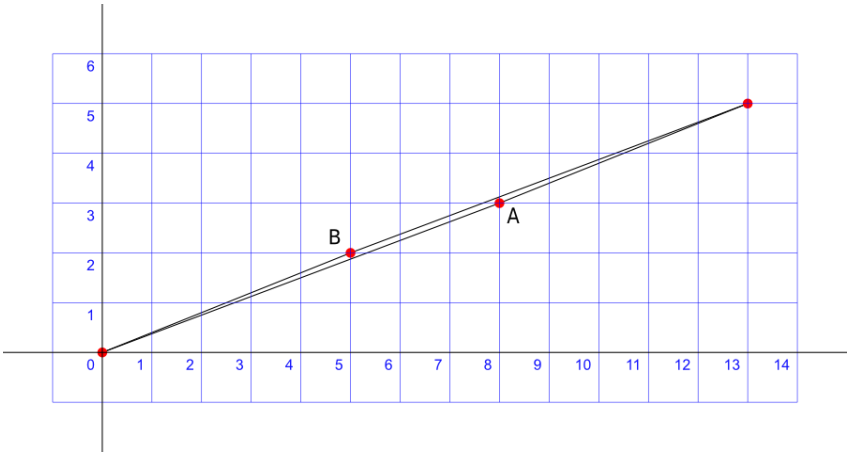
Obrázek 9 – Curryho paradox I (Gardner, 1956, s. 141)

Přeskládáním dílků B, C, D, E je získán obdélník, v němž zmizel jeden dílek. Na obrázku 10 je prázdné místo znázorněno černým čtverečkem.



Obrázek 10 – Curryho paradox II (Gardner, 1956, s. 141)

Řešení této optické iluze je znázorněno na obrázku 11.



Obrázek 11 – Curryho paradox – řešení (*Missing square puzzle*, 2017)

Úsečka, na níž leží bod A je získána diagonálním rozdělením obdélníku o rozměrech 13 × 5. Pokud jsou dílky přeskládány tak, aby vzniklo prázdné místo (viz obrázek 10), vzniká přepona trojúhelníku, která je reprezentována úsečkou s bodem B. Při přeskládání dílků dochází k „ohnutí“ přepony v důsledku odlišných vnitřních úhlů trojúhelníků C a B. Výsledný nabytý prostor je tedy kompenzován prázdným místem (*Missing square puzzle*, 2017).

1.4 Konvexní Tangramy

Následující definice popisuje konvexní útvar: „Geometrický útvar *U* (ležící na přímce, v rovině nebo v prostoru) nazýváme konvexním, jestliže úsečka spojující kterékoli jeho dva body náleží útvaru *U*.“ (Vyšín, 1964, s. 5)

Ačkoliv je z dílků *Tangramu* možné složit nepřeberné množství obrazců jejichž počet dosahuje tisíců, konvexních tvarů lze složit mnohem méně. Jak již bylo zmíněno v kapitole 1.2.1, vnitřní úhly dílků *Tangramu* jsou násobky 45 °. Konkrétně mají dílky vnitřní úhly o velikosti 45 °, 90 ° a 135 °. Z těchto tvrzení lze vyvodit, že výsledný složený polygon o *n* stranách má *x* 45° úhlů, *y* 90° úhlů a *z* 135° úhlů (Scott, 2006, s. 2-3).

Platí tedy:

$$x + y + z = n$$

$$45x + 90y + 135z = \frac{n-2}{180}$$

V následujících krocích je z první rovnice určeno *z* a poté je dosazeno do druhé rovnice.

$$z = n - x - y$$

$$45x + 90y + 135(n - x - y) = \frac{n-2}{180}$$

Úpravou je získána rovnice v následujícím tvaru:

$$2x + y = 8 - n$$

Vnitřní úhly dílků mohou být pouze kladné, platí tedy $x \geq 0$ a $y \geq 0$. Z toho vyplývá, že $n \leq 8$. Následující tabulka vyjadřuje všechny možnosti výpočtu této rovnice.

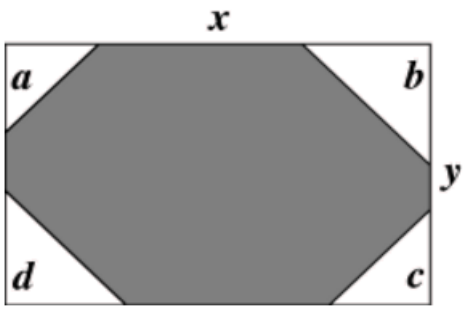
Tabulka 1 – Možné hodnoty rovnice

n	**8**	**7**	**6**	**6**	**5**	**5**	**4**	**4**	**4**	**3**
x	0	0	0	1	0	1	0	1	2	2
y	0	1	2	0	3	1	4	2	0	1
z	8	6	4	5	2	3	0	1	2	0

Tabulka ale neukazuje všechny možnosti složení *Tangramu*, tak aby byl tvar konvexní. Pro výsledek (*n, x, y, z*) = (4, 0, 4, 0) jsou už například známé dva konvexní tvary, čtverec a obdélník (Scott, 2006, s. 3).

Za předpokladu, že je každý konvexní obrazec uvnitř obdélníku, který ho obklopuje, jsou vytvořena následující pravidla:

- $x \leq y$;
- $a + b \leq x$;
- $c + d \leq x$;
- $a + d \leq y$;
- $b + c \leq y$;
- $a = min\ (a, b, c, d)$.



Obrázek 12 – Konvexní tvar (Scott, 2006, s. 3)

Za předpokladu, že má nejmenší dílek skládačky (malý trojúhelník) odvěsny o délce 1 jednotka a přeponu o délce $\sqrt{2}$ jednotek, je obsah obrazce 8 jednotek. Platí tedy rovnice (Scott, 2006, s. 4):

$$2xy - (a^2 + b^2 + c^2 + d^2) = 16$$

Spočítat všechny možnosti rovnice tak, aby byl výsledek vždy 16 a zároveň platila pravidla stanovená pro konvexní polygon, by mohlo zabrat spoustu času. Pro zjištění všech možných kombinací je využit počítačový program napsaný v jazyce Java:

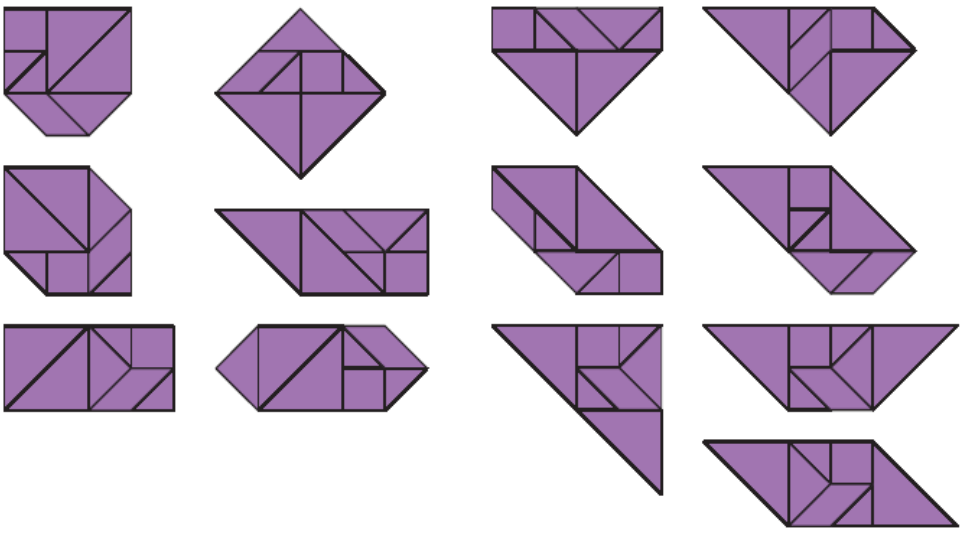
```
for (int x = 2; x <= 20; x++) {
  for (int y = 2; y <= x; y++) {
    for (int a = 0; a <= y; a++) {
      for (int b = a; b <= y; b++) {
        for (int c = a; c <= y; c++) {
          for (int d = a; d <= y; d++) {
            int m = 2 * x * y - (a * a + b * b + c * c + d * d);
            if ((a + b) <= x && (c + d) <= x  && (a + d) <= y && (b + c) <= y && m == 16) {
              System.out.println(x + " " + y + " " + a + " " + b + " " + c + " " + d);
            }
          }
        }
      }
    }
  }
}
```

Výstupem programu jsou hodnoty zobrazené v tabulce 2. Hodnoty *x* a *y* určují rozměr obdélníku, který obepíná výsledný polygon. Hodnoty *a*, *b*, *c*, *d* vyjadřují počet malých trojúhelníků *Tangramu*, které je z obdélníku potřeba odečíst na příslušných místech podle obrázku 12, aby vznikl konvexní polygon.

Tabulka 2 – Výstup hodnot z programu

Tvar	*x*	*y*	*a*	*b*	*c*	*d*	
1.	3	3	0	0	1	1	*Tangram*
2.	3	3	0	1	0	1	*Tangram*
3.	3	3	0	1	1	0	
4.	4	2	0	0	0	0	*Tangram*
5.	4	3	0	0	2	2	*Tangram*
6.	4	3	0	2	0	2	*Tangram*
7.	4	4	0	0	0	4	*Tangram*
8.	4	4	0	0	4	0	
9.	4	4	0	4	0	0	
10.	4	4	2	2	2	2	*Tangram*
11.	5	2	0	0	0	2	*Tangram*
12.	5	2	0	0	2	0	
13.	5	2	0	2	0	0	
14.	5	2	1	1	1	1	*Tangram*
15.	5	3	0	1	2	3	*Tangram*
16.	5	3	0	2	1	3	*Tangram*
17.	5	5	0	3	0	5	
18.	5	5	0	5	0	3	
19.	5	5	1	4	1	4	
20.	6	2	0	0	2	2	*Tangram*
21.	6	2	0	2	0	2	*Tangram*
22.	6	4	0	4	0	4	
23.	9	8	0	8	0	8	

Ne ze všech hodnot znázorněných v tabulce 2 je však možné složit z dílků *Tangramu* příslušné tvary. Hodnoty, ze kterých je možné *Tangram* složit, jsou v tabulce označeny. V tabulce se nacházejí ekvivalentní hodnoty. Z hodnot na 3. řádku je možné sestrojit stejný *Tangram* jako z hodnot na řádku 1, který je pouze jinak nakloněný. Stejně tak hodnoty na 8. a 9. řádku jsou ekvivalentní pro hodnoty na 7. řádku. Hodnoty na 12. a 13. řádku jsou ekvivalentní pro řádek 11 a hodnoty na řádku 18 jsou ekvivalentní pro řádek 17. Ekvivalentní tvary jsou proto vyřazeny. Čtyři tvary z tabulky s čísly 17, 19, 22 a 23 nelze z dílků *Tangramu* sestavit. Zbývá přesně 13 konvexních tvarů *Tangramu*, které je možné sestavit (Scott, 2006, s. 4-5).

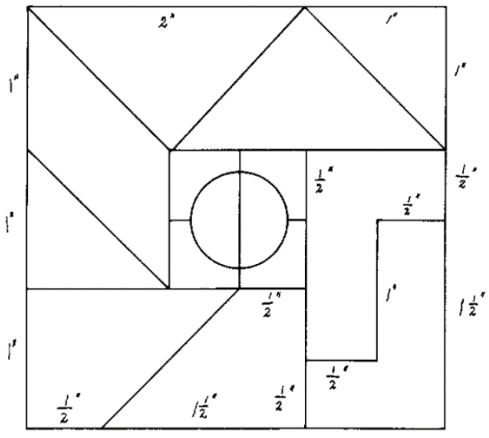


Obrázek 13 – Třináct konvexních Tangramů (Scott, 2006, s. 5)

Existence pouze třinácti konvexních tvarů (viz obrázek 13) byla poprvé dokázána v roce 1942 pány Fu Traing Wangem and Chuan-Chih Hsiungem (Scott, 2006, s. 5).

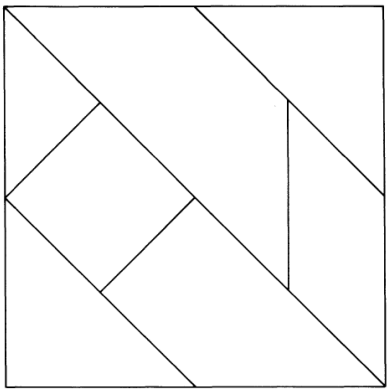
1.5 Podobné hry

Od začátku devatenáctého století, kdy byl *Tangram* velmi zpopularizován, bylo vytvořeno mnoho variací této skládačky. Jednou z nich je „15dílná skládačka“. První zmínka o tomto hlavolamu pochází z druhé poloviny devatenáctého století. Dílky mají stejně jako u *Tangramu* vnitřní úhly o velikosti 45, 90 nebo 135 stupňů. Navíc jsou tu dílky se zaoblenými hranami. S počtem patnácti dílků, lze dělat podobně komplexní obrazce jako se dvěma sadami *Tangramu* (Read, 1965, s. 81-82).



Obrázek 14 – 15dílná skládačka (Read, 1965, s. 81)

V Japonsku v roce 1742, byla vydána kniha představující skládačku velice podobnou *Tangramu*. Název knihy je možné do češtiny přeložit jako „Geniální dílky Sei Shonagon“. Sei Shonagon byla dvorní dáma na přelomu desátého a jedenáctého století. Je velice nepravděpodobné, že znala *Tangram*, přesto je skládačka *Tangramu* velice podobná. „Geniální dílky Sei Shonagon“ jsou specifické tím, že z nich lze složit čtverec dvěma možnými způsoby. Zároveň je možné složit čtverec tak, aby uvnitř byla čtvercová mezera. Ani jedna z uvedených možností není u *Tangramu* možná (Gardner, 1988, s. 97-98).



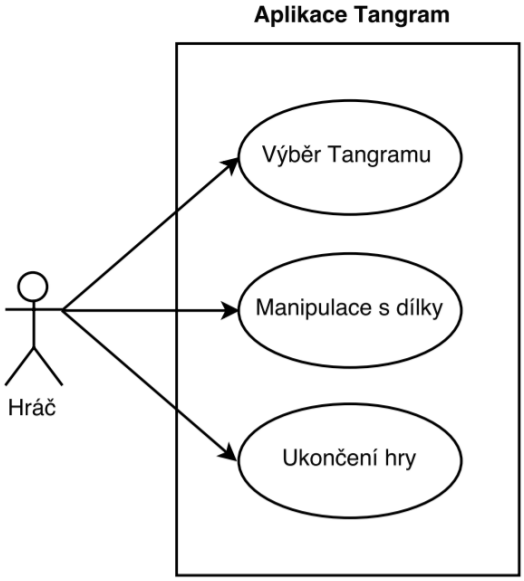
Obrázek 15 – „Geniální dílky Sei Shonagon“ (Gardner, 1988)

2 APLIKACE TANGRAM PRO PLATFORMU WINDOWS

Pro účely praktické části této bakalářské práce, byla vytvořena hra na principu logické hry *Tangram.* Aplikace je napsána v programovacím jazyce Java s využitím dalších technologií. Nejprve jsou popsány funkce aplikace, poté jsou blíže vylíčeny použité technologie pro její implementaci. V uživatelské dokumentaci je představena grafická podoba aplikace a její funkčnost. Následně jsou popsány důležité třídy aplikace a algoritmy, které aplikace využívá. Poslední podkapitola se věnuje návrhům na zlepšení aplikace.

2.1 Požadavky na aplikaci

Cílem je vytvořit aplikaci, která umí rozpoznat jakoukoliv kombinaci dílků správně umístěnou v podkladu. Hra musí dát hráči možnost výběru tvaru, který chce skládat, a následně ho zobrazit v herním okně. Dílky *Tangramu* by měly být vyobrazeny ve vhodné komponentě mimo podklad. Následně může uživatel v herním okně umisťovat dílky do vyobrazené předlohy. Hráč má možnost s dílky buď otáčet ve směru hodinových ručiček, nebo dílky otáčet zrcadlově. Během manipulace by mělo být zajištěno, aby na sebe dílky vzájemně automaticky doléhaly. Aplikace po správném složení, kterého bylo dosaženo dodržením všech pravidel hry, vyhodnotí výsledek a správně rozpozná úspěšné složení *Tangramu*. Hráč má po složení na výběr několik možností. Může pokračovat na další obrazec, který následuje, může spustit znovu hru se stejným tvarem, nebo se vrátit zpět do menu aplikace a vybrat jiný tvar.



Obrázek 16 – UML Use Case diagram

2.2 Základní charakteristika aplikace

Aplikace je naspána v programovacím jazyce Java Standard Edition 8. Vizuální část aplikace je napsána pomocí Java FX ve verzi 8. Prostřednictvím Java FX jsou řešeny všechny vizuální efekty a uspořádání grafických komponent. Jejich vzhled je definován pomocí CSS (Cascading Style Sheets). Celá aplikace je kompilována nástrojem *Apache Maven*. Pomocí něho je do projektu naimportována knihovna *XStream* sloužící pro pohodlnou práci s XML (Extensible Markup Language) soubory. Ty nesou údaje o všech polygonech, které se ve hře nachází. Aplikace se skládá ze dvou grafických oken, menu a hlavního herního okna.

2.3 Použité technologie

V této podkapitole jsou představeny technologie, použité pro vývoj aplikace *Tangram*. U jednotlivých technologií je následně popsáno, na co byly v aplikaci použity.

2.3.1 Java

Java je moderní objektově orientovaný programovací jazyk, který byl poprvé představen firmou *Sun Microsystems* v roce 1995. Podporuje všechny vlastnosti objektově orientovaného programování jako je zapouzdřenost, dědičnost a abstrakce. Jedná se o bezpečný, robustní a silně typový programovací jazyk. Podporuje také tvorbu vícevláknových aplikací, nebo je možné jazyk Java nasadit na webové stránky. Pro svůj běh potřebuje JVM (Java Virtual Machine). Díky JVM je možné stejný kód spustit na všech zařízeních, které JVM obsahují. Java je tedy multiplatformní a lehce přenosný programovací jazyk. S pomocí dalších nástrojů je možné vytvářet desktopové aplikace (Schildt, 2007, s. 11-20).

Pro účely praktické části byl vybrán jazyk Java díky všem těmto zmíněným vlastnostem.

2.3.2 Java FX

Java FX je softwarová platforma umožňující tvorbu grafických aplikací v jazyce Java. Představena byla poprvé v roce 2007 pod názvem F3. Jednalo se o skriptovací jazyk, později přejmenovaný na Java FX Script. Od verze 2 je Java FX součástí standardní knihovny jazyka Java. Ve verzi 8, která následovala po verzi 2.2, se Java FX stala náhradou pro Swing. Na rozdíl od grafické knihovny Swing nabízí Java FX spoustu výhod. Například obsahuje podporu dotykových zařízení, podporu animací nebo podporuje data binding (*JavaFX*, 2014).

Java FX byla pro aplikaci *Tangram* zvolena z důvodu jednoduchosti jejího použití, a proto, že je Java FX plnou náhradou Swingu. K tvorbě FXML souboru, který popisuje vzhled a strukturu aplikace byl využit WYSIWYG (What you see is what you get) editor *Scene Builder*.

2.3.3 CSS

Vzhled grafických komponent je v aplikaci popsán pomocí kaskádových stylů. Umožňuje oddělit vzhled a programovou část.

2.3.4 Apache Maven

Apache Maven je nástroj, který zjednodušuje a automatizuje kompilaci programu. Umožňuje import externích knihoven. *Maven* splňuje podle svých tvůrců tyto body (*Apache Maven*, 2015):

- snadný proces kompilace programu,
- jednotný systém kompilace pro všechny systémy,
- poskytnutí kvalitních a úplných informací o projektu,
- možnost přidávání nových funkcí.

Hlavním elementem *Maven* aplikace je soubor *pom.xml*, který je umístěn v kořenovém adresáři. Obsahuje všechny údaje o závislostech nebo procesu kompilace aplikace. Struktura projektu se dále dělí na balíček *java*, který obsahuje kompilovatelné *.java* soubory a balíček *resources*, který obsahuje ostatní konfigurační soubory (*.properties, .xml*).

V aplikaci *Tangram* našel *Maven* uplatnění zejména jako nástroj, umožňující jednoduchý import externí knihovny *XStream* pomocí definované závislosti.

2.3.5 XStream

Xstream je knihovna umožňující serializaci objektů do souboru typu XML a obráceně. Knihovna je distribuovaná pod licencí BSD (Berkeley Software Distribution). Jedná se tedy o jednu z nejsvobodnějších licencí pro svobodný software, která vyžaduje pouze uvedení autora a informace o licenci. Kromě práce se soubory XML umí *XStream* serializovat také do souboru typu JSON (JavaScript Object Notation). Pro aplikaci *Tangram* byla knihovna využita proto, že nabízí mnohem pohodlnější práci s XML soubory než dostupné prostředky v jazyce Java.

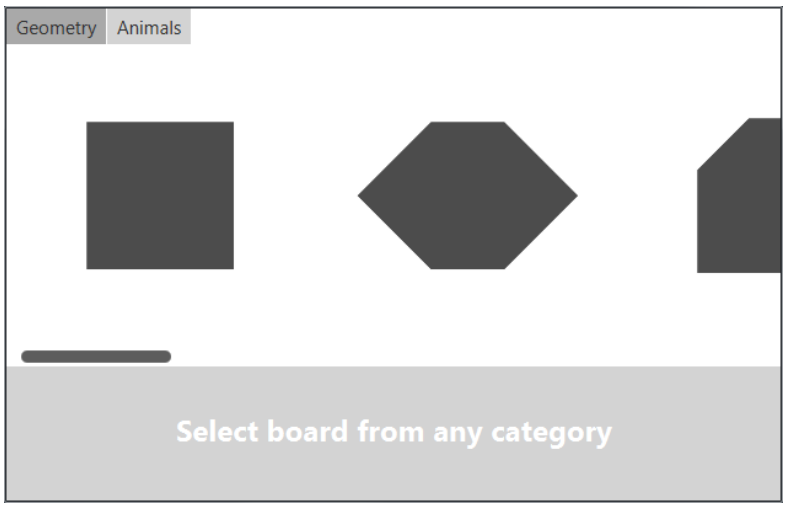
2.4 Uživatelská dokumentace

Celá aplikace se dělí na dvě hlavní části. První částí je okno obsahující menu aplikace. Uživatel zde vybírá tvar, který chce skládat. Tyto tvary jsou rozděleny do kategorií. Po vybrání jednoho tvaru může hráč spustit hru se zvoleným půdorysem *Tangramu*. Menu zmizí a objeví se druhá část aplikace, sloužící k samotné hře. Hlavní okno obsahuje herní desku, na které je zobrazen zvolený tvar pro skládání, a seznam se sedmi dílky *Tangramu*. Hráč tyto dílky přesunuje do

herní desky tak, aby složil vyobrazený tvar. Po úspěšném složení bez porušení pravidel, aplikace oznámí úspěšné složení a nabídne uživateli zopakování úrovně nebo přesun na další tvar. V následující podkapitole jsou blíže popsány všechny části aplikace.

2.4.1 Menu

Před spuštěním samotné aplikace je na malý okamžik zobrazena uvítací obrazovka, která obsahuje název aplikace a jméno autora. Po zmizení této obrazovky je načteno menu aplikace. V horní části okna menu jsou v podobě záložek vyobrazeny jednotlivé kategorie, do kterých jsou půdorysy *Tangramu* rozděleny. Ve výchozím stavu je vybrána první kategorie. Obsahem záložky je horizontální seznam s miniaturami obrazců. Pod tímto seznamem se po spuštění aplikace nachází text s výzvou pro uživatele, který vybízí k výběru některého z obrazců z jakékoliv kategorie.

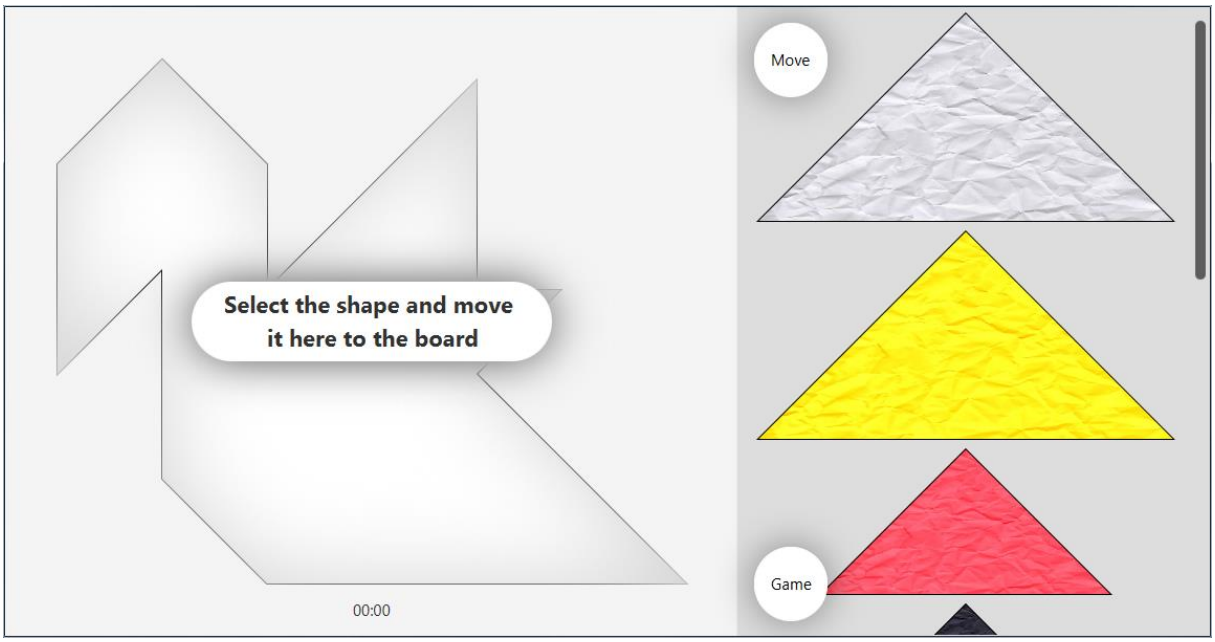


Obrázek 17 – Menu aplikace

Po zvolení tvaru výzva zmizí a namísto ní je zobrazeno tlačítko *Start*, které dovolí spustit hru s vybraným půdorysem. Pokud se hráč po výběru přepne do jiné kategorie, jeho výběr se tím ruší, tlačítko *Start* zmizí a hráč musí vybrat znovu nový tvar, aby mohl spustit hru.

2.4.2 Hlavní okno

Hlavní okno aplikace je možné rozdělit na dvě základní části. Tou první je hrací deska, na které je vyobrazen zvolený tvar, pro skládání. Kromě něho je zde zobrazena časomíra znázorňující uplynutou dobu od začátku hry. Při prvním spuštění je v této části aplikace také zobrazeno notifikační okno informující uživatele, jak aplikaci ovládat. Po první interakci s dílkem *Tangramu*, které jsou umístěné v seznamu na pravé části okna, tato notifikace zmizí.



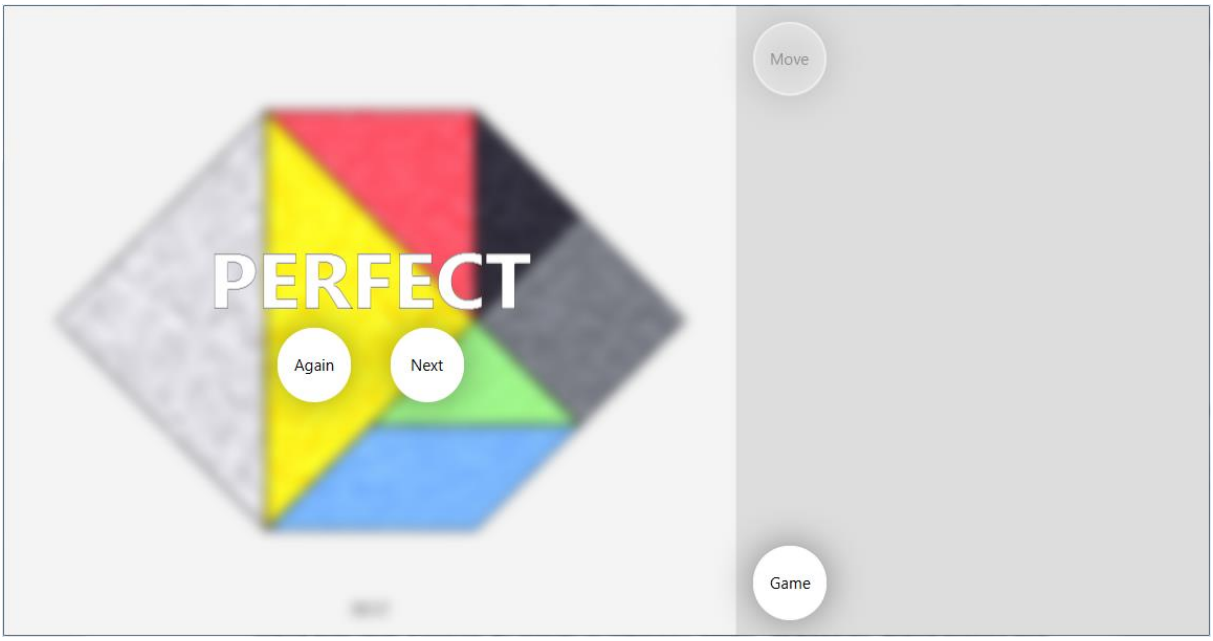
Obrázek 18 – Hlavní okno aplikace

Druhou částí hlavního okna je seznam obsahující všech sedm dílků *Tangramu*. Tyto dílky je možné přesouvat do hrací plochy, nebo je vracet z hrací plochy zpět do seznamu. Zde jsou dílky řazeny vždy ve stejném pořadí (Dva velké trojúhelníky, jeden střední trojúhelník, dva malé trojúhelníky, čtverec a rovnoběžník). V části seznamu se nacházejí dvě tlačítka s popisky *Game* a *Move*. Po stisku tlačítka *Game*, je zobrazeno pop-up menu obsahující další dvě tlačítka *New* a *Exit*. Tlačítko *New* ukončí aktuální hru a spustí novou se stejným tvarem. Tlačítko *Exit* ukončí aktuální hru a provede návrat do menu aplikace. V obou případech je zobrazeno dialogové okno, pro potvrzení akce. Tlačítko *Move* spustí pop-up menu obsahující tlačítka *Board* a *Menu*. Tlačítko *Board* přesune všechny dílky, které se nacházejí v seznamu do hrací desky. Tlačítko *Menu* přesune všechny dílky z hrací desky zpět do menu. Tyto možnosti slouží k rychlému přesouvání všech dílků z jedné části okna do druhé.

Obrázek 19 – Detail pop-up menu

Po správném složení je v části herní desky zobrazena zpráva o úspěšném složení a společně s hláškou jsou zobrazena tlačítka pro opakování té samé úrovně nebo pro pokračování na další *Tangram*. Obrazovka obsahuje dvě tlačítka, *Again* a *Next*. Tlačítko *Again* spustí znovu novou hru se stejným tvarem. Tlačítko *Next* spustí novou hru s následujícím tvarem. Nad těmito tlačítky je vyobrazen text, jehož obsah se liší podle toho, jak rychle dokázal uživatel *Tangram*

složit. Pokud se povede hráči složit *Tangram* do 30 sekund, je zobrazen text „BRILLIANT“, pokud hráč složí *Tangram* do minuty je zobrazeno „PERFECT“. Pokud hráčovi trvá složení déle jak 60 sekund, je zobrazen text „NOT BAD“.



Obrázek 20 – Obrazovka s hláškou o úspěšném složení Tangramu

2.4.3 Práce s dílky

Přesun ze seznamu do hrací desky je proveden ve dvou krocích. V prvním kroku je nutné označit dílek, který má být přesunut do hrací desky. K označení dojde stisknutím a uvolněním levého tlačítka myši (LTM) na konkrétní dílek v seznamu. Poté, ve druhém kroku, je nutné znovu stisknout LTM a tahem myši ho přesunout do hrací desky. Pro jakoukoliv další interakci s dílkem, který je již umístěn v hrací desce, stačí znovu stisknout LTM a tahem myši ho přemístit na požadované místo. Pokud je přesunut mimo hrací plochu, je automaticky navrácen do seznamu. Stiskem a uvolněním LTM na dílku bez žádného dalšího pohybu, který je umístěn v hrací desce, dojde k jeho otočení o 45 ° ve směru hodinových ručiček. Stiskem pravého tlačítka myši (PTM) je prováděno vertikální zrcadlové otáčení. Dílky si operace jako je otáčení a převracení pamatují, při přemisťování mezi seznamem a hrací deskou tedy nedochází ke ztrátě natočení. Otáčení a zrcadlové převracení fungují pouze tehdy, jsou-li dílky umístěné v hrací ploše.

Během manipulace s dílky v hrací ploše vždy dochází k jejich zarovnání do neviditelné čtvercové sítě. Po jejich uvolnění dojde k zarovnání tak, aby na sebe správně doléhaly a nepřekrývaly se. Uživatel díky tomu nemusí při skládání řešit velkou přesnost při jejich umisťování.

2.4.4 Chybové stavy

Během hry je potřeba zamezit nežádoucím chybovým stavům, které by mohly narušit její chod, nebo hru značně znepříjemnit. Aby se aplikace chovala vždy korektně, byly ošetřeny tyto stavy:

- V menu je tlačítko *Start* zobrazeno pouze tehdy, je-li vybrán tvar pro skládání. Tím je zamezeno spuštění hry bez podkladu.
- V hlavním okně je při výběru dílku ze seznamu zakázána interakce s jakýmkoliv jiným dílkem nebo pop-up menu, dokud se dílek nepřesune do hrací desky. To je zajištěno z důvodu uživatelské přívětivosti.
- Při otevření pop-up menu *Move* nebo *Game* je zakázána jakákoliv interakce s dílky, dokud nejsou menu zase zavřeny. Zavedení tohoto opatření bylo provedeno z důvodu uživatelské přívětivosti.
- Po vyhodnocení hry je vypnuto tlačítko *Move* sloužící k přesouvání dílků zpět do seznamu. Pokud by bylo tlačítko povolené, hráč by mohl i po skončení hry dílky umístit zpět do seznamu, což je nežádoucí stav.

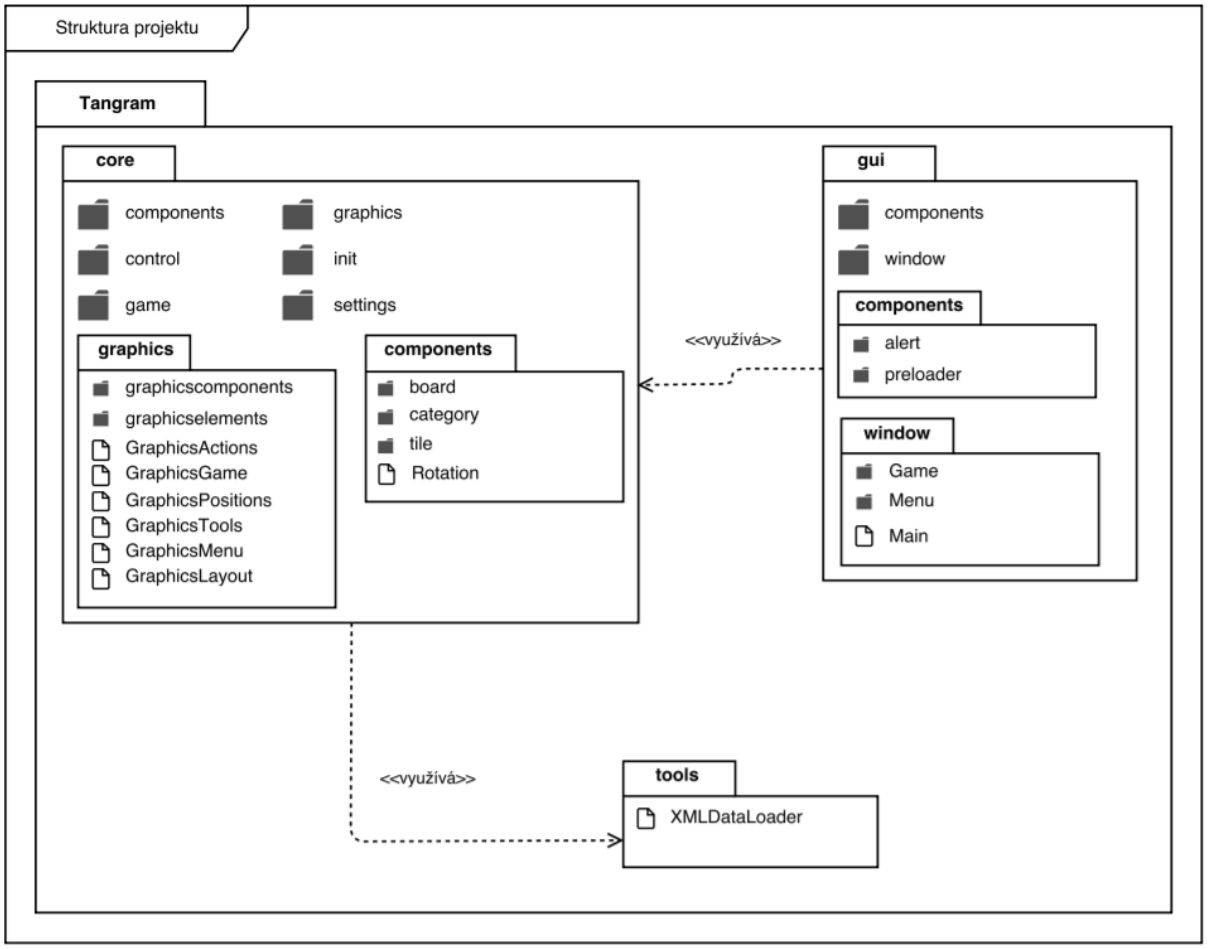
Během hry je možné také narazit na dialogová okna. Při stisku tlačítka *Exit* nebo tlačítka *New* je zobrazeno dialogové okno, se zprávou o ztracení aktuálního postupu ve hře a zda uživatel souhlasí se zvolenou možností. Stejné okno je zobrazeno, pokud uživatel nepoužije pro ukončení tlačítko *Exit,* ale použije křížek v pravém horním rohu aplikace. Při stisku tlačítka *OK* je provedena uživatelem požadovaná akce, tlačítkem *Cancel* je požadovaná akce zrušena.

Pokud hráč úspěšně složí *Tangram*, který je v menu na posledním místě a následně na obrazovce o úspěšném složení stiskne tlačítko *Next*, je uživateli zobrazeno dialogové okno s varováním, že neexistuje žádný další půdorys *Tangramu* a je přesměrován zpět do menu.

2.5 Technická dokumentace

V následující podkapitole je popsána technická stránka aplikace, struktura projektu, implementace jednotlivých částí nebo popis algoritmů, které aplikace využívá.

2.5.1 Struktura projektu



Obrázek 21 – UML diagram projektu

Projekt je rozdělen do tří základních balíčků *core*, *gui* a *tools*, viz obrázek 21. Balíček *core* obsahuje veškerou logickou část aplikace. V balíčku *core.components.board* se nachází třída *Board*, která reprezentuje výsledný složený tvar *Tangramu*. Třída Tile, umístěná v balíčku *core.components.tile*, představuje jeden z dílků Tangramu. Uvnitř balíčku *core.control* se nachází třída *Control*, která obsahuje statické metody, jejichž návratový typ je typu *boolean*. Metody třídy *Control* slouží v aplikaci ke zjištění a vyhodnocování stavů. Třída *Game* umístěná v balíčku *core.game* obsahuje kolekci všech dílků *Tangramu*. Slouží k vyhodnocení hry a její inicializaci. Balíček *core.graphics* obsahuje třídy pro veškerou práci s grafickými prvky v aplikaci. Nachází se zde například třídy *BoardCell* a *TileCell*, pomocí kterých je upraven vzhled položek seznamu tak, aby zobrazovaly konkrétní polygon. Třída *GraphicsActions*

definuje operace pro práci s dílky jako je přesouvání, otáčení nebo zrcadlové převracení. Třída *GraphicsPositions* definuje statické metody pro zarovnání dílků v hracím poli a třída *GraphicsTools* obsahuje převážně metody sloužící k matematickým operacím, jako je zjištění vzdálenosti dvou bodů, nebo zjištění souřadnic středu polygonu. Dále se v balíčku *core.init* nachází třída *Initialization,* provádějící inicializaci všech tříd. V balíčku *core.settings* je umístěna třída *Configuration*, která obsahuje konstantní hodnoty.

V balíčku *gui.components* je třída *Alerts* obsahující statické metody pro vyvolání dialogových oken. V balíčku *gui.preloader* se nachází spouštěcí třída pro uvítací obrazovku aplikace. V balíčku *gui.window* se nacházejí třídy pro obsluhu grafického rozhraní aplikace. Je zde také umístěna spouštěcí třída pro celou aplikaci.

Balíček *tools* obsahuje Třídu *XMLDataLoader*, která definuje metody pro serializaci dat ze XML souborů.

2.5.2 Struktura XML dat

Údaje o všech polygonech, které se ve hře vyskytují (dílky, podklady) jsou uloženy v XML souborech. Aplikace obsahuje dva takové soubory a jsou umístěné v balíčku *resources.data*. První soubor s názvem *boards.xml* nese údaje o souřadnicích pro podklady. Druhý s názvem *tiles.xml* obsahuje souřadnice dílků *Tangramu* při jednotlivých úhlech. Z dat uložených v těchto souborech jsou data deserializována a v aplikaci jsou na základě těchto dat vytvářeny příslušné objekty, reprezentující podklady, nebo dílky *Tangramu*. V případě souboru *tiles.xml* je kořenovým prvkem element *tiles*. Ten obsahuje elementy *tile* s atributy nesoucími informaci o typu a rotaci dílku. Uvnitř tohoto elementu se nachází další element *imagePath*, který nese údaje o cestě k obrázku, sloužícího jako pozadí pro dílek. Hlavní údaje nesou elementy *points* a *flippedPoints,* jelikož obsahují souřadnice dílků. V následující části je zobrazena struktura souboru tiles.xml, konkrétně jsou zobrazena data pro jeden dílek.

tiles.xml

```
<tiles>
    <tile tileType="LARGE_TRIANGLE_1" rotation="ANGLE_0" flippedRotation="ANGLE_0">
        <imagePath>graphics/textures/tile/white.jpg</imagePath>
        <points>
            <double>200.0</double>
            <double>0.0</double>
            <double>400.0</double>
            <double>200.0</double>
            <double>0.0</double>
            <double>200.0</double>
        </points>
        <flippedPoints>
            <double>200.0</double>
            <double>0.0</double>
            <double>400.0</double>
            <double>200.0</double>
            <double>0.0</double>
            <double>200.0</double>
        </flippedPoints>
    </tile>
<tiles>
```

Kořenovým prvkem souboru *boards.xml*, který nese údaje o podkladech *Tangramu* je element *boards*. V něm se nachází prvky *board* s atributy o jejich typu a příslušné kategorii. Element *points* obsahuje souřadnice polygonu. V následující části je zobrazena část souboru *boards.xml*.

boards.xml

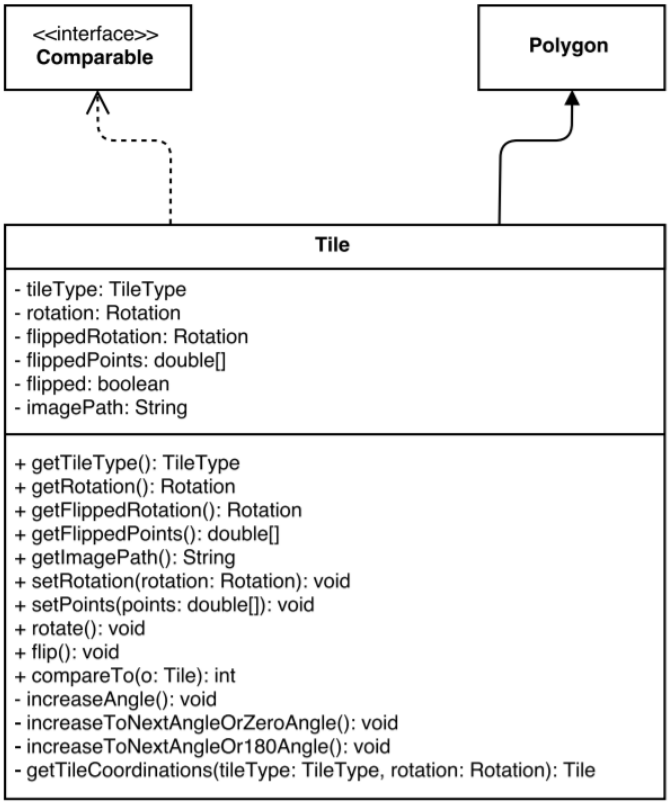
```
<boards>
    <board boardType="BOARD_1_GEOMETRY" categoryType="GEOMETRY">
        <points>
            <double>0.0</double>
            <double>0.0</double>
            <double>400.0</double>
            <double>0.0</double>
            <double>400.0</double>
            <double>400.0</double>
            <double>0.0</double>
            <double>400.0</double>
        </points>
    </board>
<boards>
```

2.5.3 Důležité třídy

V této podkapitole jsou popsány důležité třídy aplikace a je blíže popsána jejich implementace.

Třída Tile

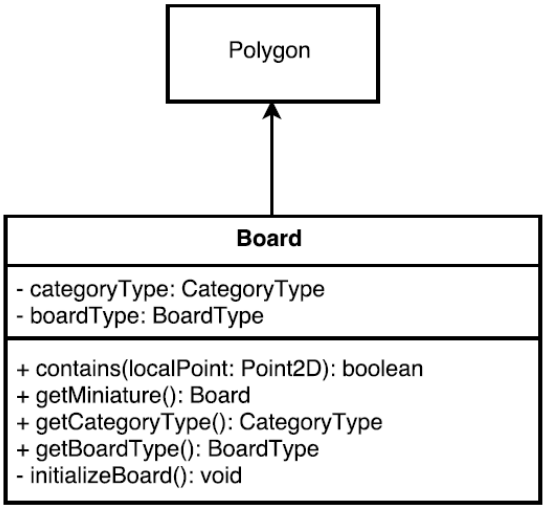
Třída Tile představuje v aplikaci dílek skládačky *Tangram*. Je potomkem vestavěné třídy *Polygon*, která je součástí Java FX. Zároveň třída implementuje rozhraní *Comparable*, aby bylo možné dílky v kolekci řadit. Mezi nejdůležitější metody, které třída poskytuje, patří metoda *rotate* sloužící k otočení dílku o 45 ° ve směru hodinových ručiček, nebo metoda *flip* pro vertikální zrcadlové otočení dílku.



Obrázek 22 – UML diagram třídy Tile

Třída Board

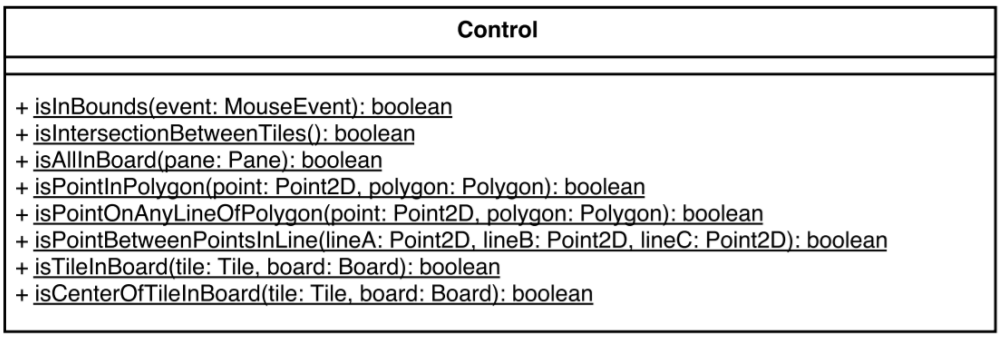
Stejně jako třída *Tile* je třída *Board* potomkem vestavěné Java FX třídy *Polygon*. Třída v aplikaci představuje výsledný obrys, který se z dílků v aplikaci skládá. Nese informaci o tom, jakého je typu a do jaké spadá kategorie. Nejdůležitější metoda této třídy je metoda s názvem *contains*, která vyhodnocuje, zda bod se souřadnicemi *x* a *y*, zadaný v parametru metody, leží uvnitř polygonu, který třída *Board* reprezentuje.



Obrázek 23 – UML diagram třídy Board

Třída Control

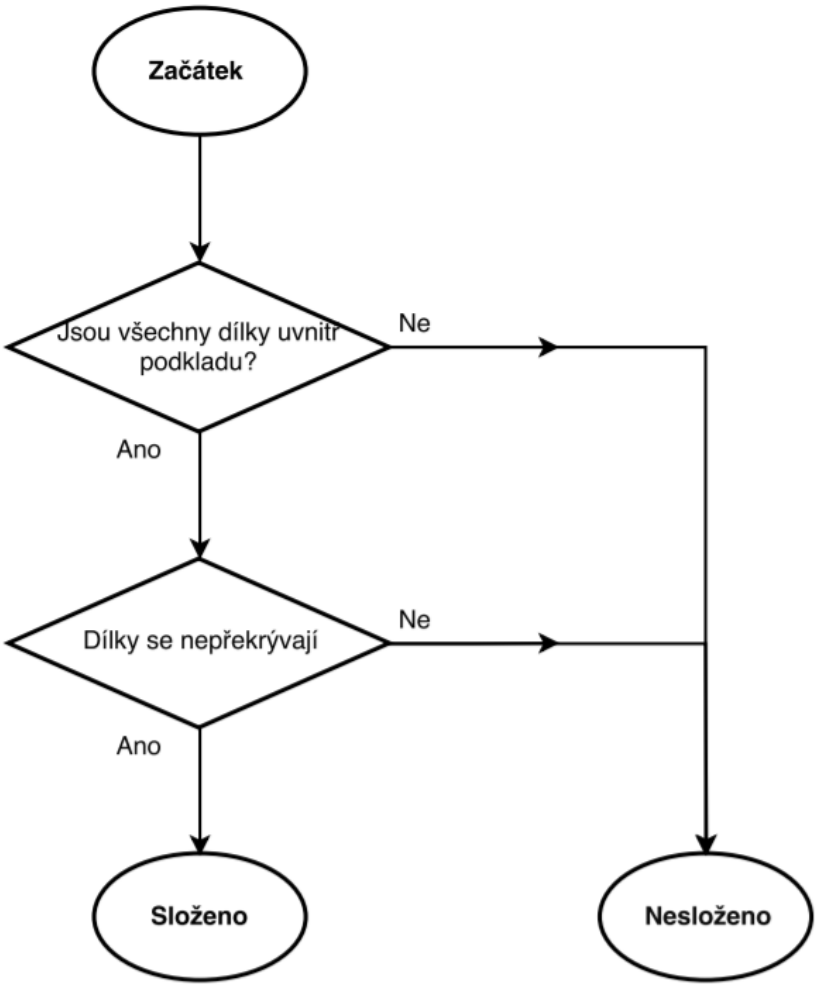
Třída nemá žádné atributy a díky privátnímu konstruktoru není možné vytvářet její instance. Obsahuje pouze statické metody, jejichž návratový typ je typu *boolean*. Třída *Control* slouží k zjišťování stavů v aplikaci. Metoda *isInBounds* vyhodnocuje, zda je dílek uvnitř hrací desky. Všechny ostatní metody slouží k vyhodnocení, zda je dílek uvnitř předlohy *Tangramu* a zda jsou dílky uvnitř předlohy správně umístěny. Metoda *isIntersectionBetweenTiles* ověřuje vzájemné překrývání mezi dílky.



Obrázek 24 – UML diagram třídy Control

2.5.4 Algoritmus vyhodnocení

V následující části je popsáno, jak probíhá vyhodnocení správného umístění dílků v podkladu. Ke každé části je popsána i její implementace prostřednictvím programovacího jazyka. Algoritmus musí být schopen rozpoznat jakoukoliv kombinaci dílků, správně umístěnou uvnitř podkladu. Zároveň musí být dodržena všechna pravidla Tangramu. Algoritmus tedy ověřuje, zda jsou všechny dílky umístěné uvnitř podkladu a zda se žádný z dílků nepřekrývá.



Obrázek 25 – Vývojový diagram algoritmu pro vyhodnocení správnosti složení Tangramu

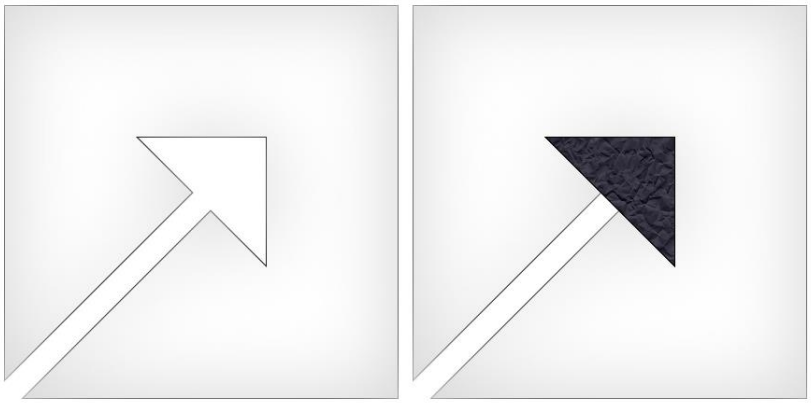
Nejprve je zjištěno, zda nedochází k překrývání mezi dílky. To je zajištěno pomocí statické metody třídy *Control* s názvem *isIntersectionBetweenTiles*. Ve dvou cyklech je zde procházeno všech sedm dílků a pro každou jejich možnou dvojici je ověřeno, zda se dílky nepřekrývají. Metoda vrací hodnotu *true* pokud se alespoň dva dílky překrývají.

```
1  public static boolean isIntersectionBetweenTiles() {
2    for (Tile tile1 : Game.getListOfTiles()) {
3      for (Tile tile2 : Game.getListOfTiles()) {
4        if (!tile1.equals(tile2) && !Shape.intersect(tile1, tile2).getBoundsInLocal().isEmpty()) {
5          return true;
6        }
7      }
8    }
9    return false;
10 }
```

V dalším kroku je v cyklu vyhodnoceno, zda se nachází všech sedm dílků uvnitř podkladu. Toho je dosaženo tak, že jsou procházeny všechny jednotlivé body vrcholů dílku, a pro tento každý bod je metodou *contains* vyhodnocováno jeho umístění vůči podkladu. Toto je implementováno za pomocí metody *isTileInBoard*.

```
1  private static boolean isTileInBoard(Tile tile, Board board) {
2    List<Point2D> tilePoints = GraphicsTools.getPointsLocalToParent(tile);
3      for (Point2D point : tilePoints) {
4        if (!board.contains(point)) {
5          return false;
6        }
7      }
8    return true;
9  }
```

Kromě vyhodnocení všech rohových bodů polygonu je nutné pro speciální případy kontrolovat, zda se uvnitř podkladu nachází i střed polygonu. Metoda *contains*, která se nachází ve třídě *Board* volá dvě metody ze třídy *Control*. Metoda *isCenterOfTileInBoard* zjišťuje, zda se střed polygonu nachází uvnitř podkladu. Někdy může dojít k situaci, kdy jsou všechny hrany dílku uvnitř podkladu, ale střed dílku se nachází mimo podklad. Příklad je vyobrazen na obrázku 26. V takové situaci, bez implementace této funkce, by algoritmus vyhodnocení dospěl k závěru, že se dílek nachází uvnitř podkladu. To je nežádoucí stav. V praxi však může stále nastat, že je dílek svými vrcholy i středem uvnitř polygonu, a přesto je nějaká jeho část mimo podklad. Pokud dojde k takové situaci, je sice dílek chybně vyhodnocen, že je uvnitř podkladu, ale nenaruší tím správný chod hry. Stále je totiž nutné poskládat všech sedm dílků dovnitř, bez jakéhokoliv vzájemného překrývání mezi sebou. Případy chybného vyhodnocení dílku v aplikaci jsou velice minimální, a pokud nastanou, neovlivní její správný chod.



Obrázek 26 – Krizová situace

Druhou metodou, kterou volá metoda *contains* je metoda *isPointInPolygon*. Ta zjišťuje přítomnost bodu, předaného v parametru metody, uvnitř polygonu. V následující části je příklad implementace této metody.

```
1  public static boolean isPointInPolygon(Point2D localPoint, Polygon polygon) {
2    List<Point2D> points = GraphicsTools.getPointsLocalToParent(polygon);
3      boolean result = false;
4      for (int i = 0, j = points.size() - 1; i < points.size(); j = i++) {
5        double xi = points.get(i).getX();
6        double xj = points.get(j).getX();
7        double yi = points.get(i).getY();
8        double yj = points.get(j).getY();
9        if (((yi > localPoint.getY()) != (yj > localPoint.getY())) &&
                  (localPoint.getX() < (xj - xi) * (localPoint.getY() - yi) / (yj - yi) + xi)) {
10         result = !result;
11       }
12     }
13   return result;
14 }
```

Tato metoda funguje spolehlivě při určení, zda je bod uvnitř polygonu, který představuje půdorys *Tangramu*. Pokud je však bod přesně na hraně půdorysu, metoda ho vyhodnotí jako bod, který neleží uvnitř polygonu. Z tohoto důvodu byla implementována další metoda s názvem *isPointOnAnyLineOfPolygon*, která vyhodnocuje, zda bod leží na jakékoliv hraně polygonu.

```
1  public static boolean isPointInOnAnyLineOfPolygon(Point2D point, Polygon polygon) {
2    List<Point2D> points = GraphicsTools.getPointsLocalToParent(polygon);
3      for (int i = 0, j = points.size() - 1; i < points.size(); j = i++) {
4        if (isPointBetweenPointsInLine(points.get(j), points.get(i), point)) {
5          return true;
6        }
7      }
8    return false;
9  }
```

Metoda prochází všechny dvojice sousedních vrcholů polygonu a pomocí další metody *isPointBetweenPointsInLine* zjišťuje, zda mezi nimi leží bod předaný v parametru metody. Jsou tedy vybrány vždy dva vrcholy polygonu. Nechť jsou tyto body označena A a B a nechť bod X je bod ležící mezi body A a B. Ověření, zda leží X na linii přesně mezi A a B, tzn. na hraně polygonu, je provedeno následovně:

$$|AB| = |AX| + |BX|$$

Následuje příklad implementace metody *isPointBetweenPointsInLine*.

```
1 private static boolean isPointBetweenPointsInLine(Point2D lineA, Point2D lineB, Point2D point) {
2   return GraphicsTools.distance(lineA, point) + GraphicsTools.distance(lineB, point) ==
                GraphicsTools.distance(lineA, lineB);
3 }
```

2.6 Návrh na zlepšení aplikace

V následující podkapitole jsou popsány návrhy, jak by mohla být aplikace dále rozšířena, aby nabízela větší funkčnost nebo lepší uživatelskou přívětivost.

Pokaždé, když uživatel skládá *Tangram*, vidí půdorys, do kterého má dílky umisťovat. Možným rozšířením aplikace je neviditelnost půdorysu, kdy uživatel ví, jaký má skládat tvar, ale dílky umisťuje do volné plochy tak, aby výsledná kombinace dílků představovala zvolený tvar *Tangramu*. Nemusí je umisťovat na předem určené místo.

V celé aplikaci chybí možnost jakékoliv konfigurace. Chybí možnost nastavení velikosti hrací desky a velikosti dílků, nebo možnost nastavení jejich vzhledu. V aplikaci také chybí zvukové efekty.

Pokud hráč *Tangram* složí, algoritmus sice správně vyhodnotí, že je *Tangram* složený, ale aplikace si tuto skutečnost nezapamatuje. Při novém spuštění hry, tedy hráč neví, které *Tangramy* už složil a které ne.

3 ZÁVĚR

Cílem práce bylo popsat logickou hru *Tangram* a následně hru implementovat v programovacím jazyce pro platformu Windows. V teoretické části byla představena historie a původ názvu této hry. Byla objasněna pravidla a způsoby, jakými *Tangram* hrát. Další kapitola se zabývala zvláštnostmi *Tangramu*, mezi které patří hlavně optické iluze. Dále byl představen problém konvexních *Tangramů* a byly vylíčeny hry podobné *Tangramu*. V praktické části práce byla provedena implementace aplikace *Tangram* pro platformu Windows.

Aplikace se skládá ze dvou oken. Po spuštění je zobrazeno menu hry, kde hráč vybírá tvar, který chce skládat. Tvary složeného *Tangramu* jsou v menu rozděleny do kategorií. Po spuštění hry je zobrazeno druhé okno hry, které obsahuje půdorys *Tangramu* a dílky pro jeho skládání. Hráč tyto dílky umisťuje do vyobrazeného tvaru. Aplikace umí rozpoznat libovolnou kombinaci dílků správně umístěnou uvnitř podkladu a oznámit správný výsledek uživateli. Po složení může hráč opakovat skládání stejného tvaru, nebo může pokračovat na další půdorys *Tangramu*, který následuje.

Praktická textová část se věnovala aplikaci jak z uživatelského, tak z technického hlediska. V uživatelské dokumentaci bylo popsáno grafické rozhraní aplikace a práce s aplikací. Byly charakterizovány technologie využité při její implementaci. Technická dokumentace se věnovala struktuře projektu a struktuře XML souborů, ve kterých jsou uloženy informace o dílcích a podkladech. Také byly formulovány důležité třídy aplikace.

Pro řešení správnosti složení *Tangramu* byl navržen a implementován vlastní algoritmus. Ten vyhodnocuje polohu všech dílků vzhledem k polygonu, který reprezentuje půdorys *Tangramu*. Konkrétně je pro každý dílek skládačky zjištěna poloha jeho vrcholů a středu vůči podkladu. Může nastat situace, že jsou všechny vyhodnocované body dílku uvnitř podkladu, ale přesto dílek nějakou částí podklad přesahuje. Takové situace mohou nastat velice ojediněle a pokud se tak stane, neovlivní správný chod algoritmu, protože druhým kritériem, podle kterého se algoritmus rozhoduje, je vzájemné překrývání mezi dílky. Pokud by byl dílek umístěn tak, aby jeho vrcholy a střed byly uvnitř podkladu, a i přesto by svou částí přes něj přesahoval, znamenalo by to, že je umístěn velice nestandardně a překrýval by se s jiným dílkem. Algoritmus je tedy spolehlivý a vždy správně určí konečné složení *Tangramu*.

POUŽITÁ LITERATURA

Apache Maven. In: Wikipedia: the free encyclopedia, 2015 [online]. San Francisco (CA): Wikimedia Foundation, [cit. 2017-05-09]. Dostupné z: https://cs.wikipedia.org/wiki/Apache_Maven

DUDENEY, Henry Ernest, 2007. *Good Old-Fashioned Challenging Puzzles and perplexing mathematical problems.* Chichester: Summersdale Publishers. ISBN 1-84024-557-3.

GARDNER, Martin, 1956. *Mathematics, magic and mystery*. New York: Dover. ISBN 0-486-20335-2.

GARDNER, Martin, 1988. *Time travel and other mathematical bewilderments*. New York: W.H. Freeman. ISBN 0716719258.

JavaFX. In: Wikipedia: the free encyclopedia, 2014 [online]. San Francisco (CA): Wikimedia Foundation. [cit. 2017-05-09]. Dostupné z: https://cs.wikipedia.org/wiki/JavaFX

Legendy o vzniku tangramu: Císař Yu a keramická dlaždice. 2012 [online]. [cit. 2017-04-23]. Dostupné z: http://mojehlavolamy.webnode.cz/news/legendy-o-vzniku-tangramu-cisar-yu-a-keramicka-dlazdice/

LOYD, Sam a Peter. VAN NOTE, 2007. *Sam Loyd's book of tangrams*. Mineola, N.Y.: Dover. ISBN 0-486-45424-x.

Missing square puzzle. In: Wikipedia: the free encyclopedia, 2017 [online]. San Francisco (CA): Wikimedia Foundation. [cit. 2017-04-23]. Dostupné z: https://en.wikipedia.org/wiki/Missing_square_puzzle

READ, Ronald C., 1965. *Tangrams: 330 puzzles*. New York: Dover Publications. ISBN 0-486-21483-4

Rozměry tanů: dílků Tangramu, 2012 [online]. [cit. 2017-04-23]. Dostupné z: http://mojehlavolamy.webnode.cz/news/rozmery-tanu-dilku-tangramu1/

SCHILDT, Herbert, 2007. *Java the complete reference*. 7th ed. New York: McGraw-Hill. ISBN 9780071631778.

SCOTT, Paul. Convex Tangrams. *The Australian Mathematics Teacher*. 2006, 62(2), 2-5. ISSN 0045-0685.

VYŠÍN, Jan, 1964. *Konvexní útvary*. Praha: Mladá fronta. Škola mladých matematiků, sv. 9.

ZAPLETAL, Miloš, 1983. *Kniha hlavolamů*. Praha: Albatros.