**Дніпровський ліцей інформаційних технологій**

**при Дніпровському національному університеті**

**імені Олеся Гончара**

**Випускна робота**

**на тему:**

**Створення комплексу задач для Турінтелу**

**Виконавець:**

**ліцеїстка ІІІ-В-2 курсу**

**Виноградова Аліса**

**Керівник роботи:**

**Боровик Л.І.**

**Дніпро**

**2017**

ЗМІСТ

[ЗМІСТ 2](#_Toc499584041)

[ВСТУП 3](#_Toc499584042)

[ОСНОВНА ЧАСТИНА 4](#_Toc499584043)

[Теоретична частина 4](#_Toc499584044)

[Теоретична частина з інформатики 6](#_Toc499584045)

[Опис роботи 9](#_Toc499584046)

[Структурна схема програми 18](#_Toc499584047)

[Програмні та апаратні вимоги 19](#_Toc499584048)

[Комплектація програми 19](#_Toc499584049)

[Використані програмні засоби 19](#_Toc499584050)

[ВИСНОВОК 20](#_Toc499584051)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 21](#_Toc499584052)

ВСТУП

Турінтел – це конкурс, відомий учням усієї області, і по сумісництву те, чим Ліцей інформаційних технологій може справді пишатися, адже він проходить кожного року саме у межах цього навчального закладу.

Конкурс проходить в електронному вигляді, і тому темою нашої кваліфікаційної роботи є створення гідної програми з різноманітними завданнями, які були б цікавими для розв’язання і визначали рівень розвитку логіки учнів, що проходять тестування.

Актуальність цієї роботи полягає у тому, що Турінтел проходить у ЛІТі кожного року, тож нашу працю будуть використовувати зараз і ще роки після того, як ми завершимо навчання.

Час від часу треба його оновлювати, переглядати деякі задачі, перевіряти їх актуальність, видаляти чи додавати деякі.

Звичайно, практичне застосування є головною метою розробки комплексу задач, але ми також зацікавлені у підбиранні самих завдань, адже ми колись теж приймали участь у Турінтелі і знаємо, якими саме потрібні бути завдання для максимальної об’єктивності балу за тестування.

Логічні завдання можна віднести до своєрідної гімнастики для розуму. Рішення таких завдань розвиває логічне та нестандартне мислення, кмітливість, пам'ять, які необхідні кожній людині. Завдання на логіку бувають різними. Наприклад, математичними, з підступом, на кмітливість, у вигляді головоломок.

Десять завдань, що є основою нашої роботи, допомагають розкрити та оцінити такі навички школярів, як здатність логічно та аналітично мислити, а також застосовувати свої знання з математики та української мови.

Більш того, цей проект допоміг нам поширити свої знання о програмуванні мовою C# та навчитися самостійно знаходити та опрацьовувати потрібну інформацію, що, звісно, є дуже корисним досвідом, який знадобиться і в майбутньому житті.

ОСНОВНА ЧАСТИНА

## Теоретична частина

Моя частина роботи складається з п’яти задач: «Таблиця Шульте», «Слова», «Числові ряди», «Дві однакові», «Знайти предмети».

1. «Таблиця Шульте»

Розроблено спочатку як психодіагностичну пробу для дослідження властивостей уваги німецьким психіатром і психотерапевтом Вальтером Шульте (нім. Walter Schulte; 1910-1972).

«Таблиця Шульте» – це таблиця, із випадково розміщеними числами в клітинках. Таблиці бувають різними, але стандартні – 5 рядів та 5 стовпців. Потрібно по черзі їх відшукати. Шукати бажано дивлячись в центр квадрата, а не бігати очима за кожною цифрою.

За допомогою таких таблиць можна тренувати та розвивати увагу, периферійний (бічний) зір – один з найбільш важливих інструментів швидкості читання. Здійснюючись периферійними областями сітківки, воно дозволяє замість декількох букв бачити і якісно сприймати слово або навіть рядок цілком. Таким чином розвинене бічний зір скорочує час сприйняття інформації і безумовно приносить користь, розвиваючи швидкість читання.

Іноді, за допомогою таблиці Шульте перевіряють потенційних працівників деякі рекрутингові агентства, в основному, для посад, які потребують високої концентрації уваги. Крім того, за допомогою таблиць Шульте дуже зручно перевірити як ви включаєтесь в роботу. Адже є люди, котрі повинні одразу братись за складні завдання, на самому початку дня, а іншим краще почати із легших завдань, щоб «розігнатись», та привести психіку в «стан готовності».

1. «Слова»

У завданні представлений список різних слів. Через 10 секунд ці слова змішуються, але до них додається ще одне слово, яке потрібно знайти. Це завдання повторюється тричі.

Ця задача перш за все перевіряє пам’ять та увагу.

1. «Числові ряди»

У цих завданнях необхідно розгадати принцип, за яким задається певна послідовність, і продовжити її. Часто буває так, що послідовність можна продовжити різними способами, і під кожен з них підібрати закономірність, за якою вона будувалася. Ряди будуються на різних математичних закономірностях з використанням різних дій: додавання, віднімання, множення, ділення, піднесення до ступеня, добування кореня та інше.

Рішення логічних задач на послідовності допомагає в пошуках відповіді вийти за рамки. Той, кому це легко вдається, знайде найбільше варіантів.

1. «Дві однакові»

У цій задачі зібрано різні картки із зображенням різнобарвних литсочків. Кожна картка має свою пару, під час гри її потрібно знайти. Для цього учню потрібно вибрати спочатку одну картку, потім другу, якщо вони збіглися, то ці картки більше не закриваються, а гравець на один крок стає ближче до виграшу. Якщо ж він не вгадав, то картки знову перегортаються, але залишаються на своїх місцях.

Гра розвиває пам'ять, образне мислення та уважність.

1. «Знайти предмети»

Метою цієї задачі є знайдення предметів зі списку на малюнку. Такі задачі часто використовуються в іграх, зазвичай в аркадах, або як проміжне завдання у стратегіях.

## Теоретична частина з інформатики

Робота виконана у програмному середовищі Microsoft Visual Studio 2013 мовою C#. Для редагування картинок, використаних у задачах, були використані наступні програми: Paint, Gimp 2.

Для моєї роботи я використала такі елементи управління:

1. ***Label*** – для відображення різноманітної текстової інформації.
2. ***Panel*** – для групування різних елементів управління.
3. ***Timer*** – для відстеження часу, затраченого на проходження задачі, а також роботи деяких функцій.
4. ***Button*** – для початку або завершення виконання задачі.
5. ***PictureBox*** – для виведення графічної інформації моєї роботи.
6. ***TextBox*** – для вводу інформації (відповідей) учнем.
7. ***ComboBox*** – для вибору класу учня.

Як вже зазначалося, моя частина проекту складається з п’яти задач: «Таблиця Шульте», «Слова», «Числові ряди», «Дві однакові», «Знайти предмети». Також я робила форми «Головна» та «Задачі».

У більшості своїх задач я створюю масиви таких елементів управління, як ***label*** та ***picturebox***. Зокрема, масиви ***label*** я використовую у задачах «Знайди предмети», «Таблиця Шульте» та «Слова», а масив ***picturebox*** – у задачі «Дві однакові». Для кожного такого елемента я вказую розміри та його положення на формі (зазвичай у циклі). Наприклад (з задачі «Таблиця Шульте»):

void LblCreate()

{

for (int i = 0; i < 49; i++)

{

labels[i] = new Label();

labels[i].AutoSize = false;

labels[i].Width = 80;

labels[i].Height = 80;

labels[i].TextAlign = label2.TextAlign;

labels[i].BackColor = System.Drawing.Color.FloralWhite;

labels[i].Click += new System.EventHandler(this.labClick);

}

for (int i = 0; i < 7; i++)

{

l = 5;

for (int j = 0; j < 7; j++)

{

labels[p].Top = n;

labels[p].Left = l;

l += 85;

p++;

}

n += 85;

}

this.panel1.Controls.AddRange(this.labels);

}

Також, у всіх моїх задачах використовується однакова функція *Stop*:

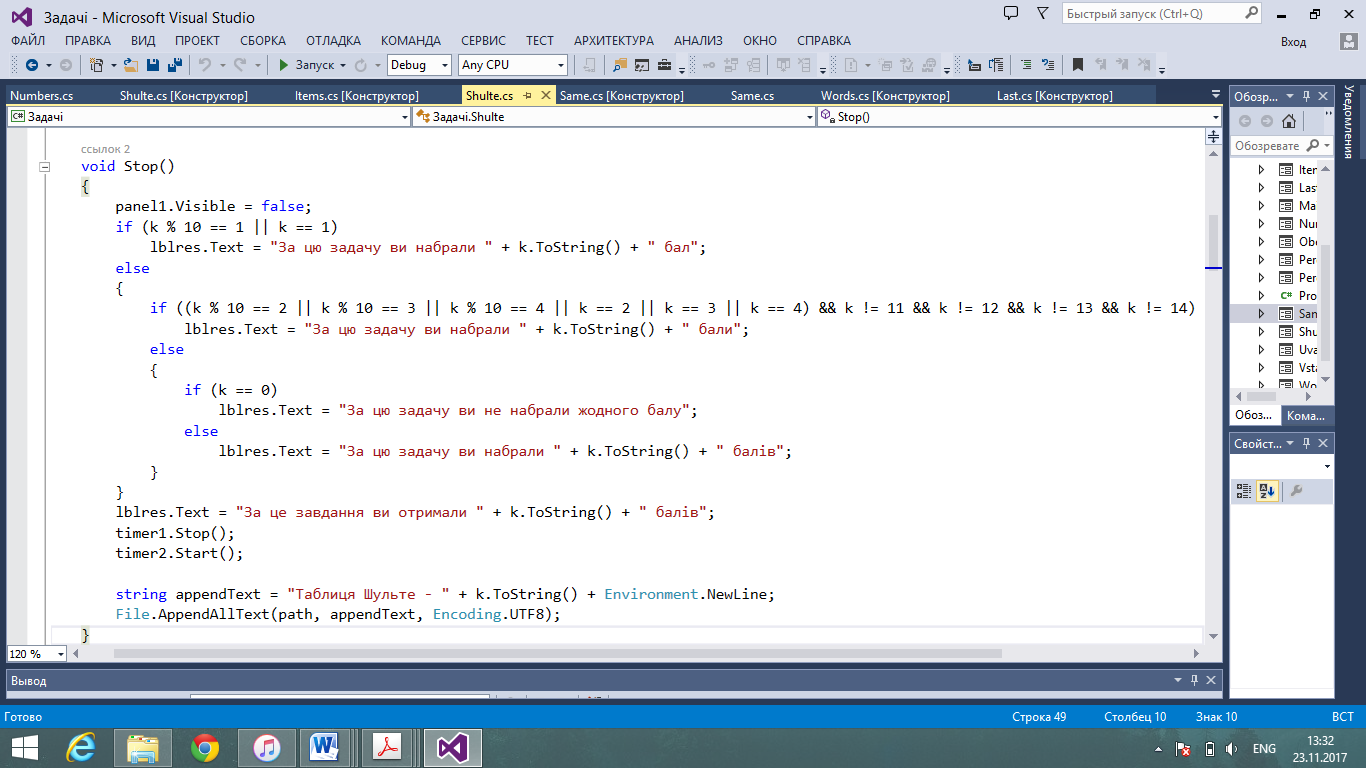


Рисунок 1.Функція "Stop"

До цієї функції програма звертається, якщо час вийшов (перевіряється у ***timer***) або при натисканні на кнопку «Закінчити». Тоді запускається ще один ***timer***, який відраховує три секунди перш ніж знову відкрити форму «Задачі» для вибору наступного завдання.

Щоб учень міг побачити, скільки часу вже витрачено на задачу, він виводиться на формі у кожній задачі по події *timer2\_Tick*:

private void timer2\_Tick(object sender, EventArgs e)

{

if (t > 0)

{

t--;

tm = t / 60;

ts = t % 60;

if (ts >= 10)

lbltime.Text = tm.ToString() + ":" + ts.ToString();

else

lbltime.Text = tm.ToString() + ":0" + ts.ToString();

}

else

Stop();

Ще один цікавий фрагмент, який присутній у декількох задачах, використовується, коли нам потрібно заповнити масив випадковими числами, але так, щоб вони не повторювалися. Для цього я використовую функцію *Choose*.

void Choose()

{

int x, y;

Random rand = new Random();

for (int i = 1; i <= 49; i++)

{

q[i - 1] = i;

}

for (int i = 0; i < 49; i++)

{

x = i + rand.Next(49 - i);

y = q[x];

q[x] = q[i];

q[i] = y;

}

}

## Опис роботи

Запускаючи програму учень потрапляє на головну форму, де він повинен записати своє ім’я та фамілію. Якщо це не буде зроблено, то він не зможе піти далі і побачить помилку, яка буде виведена за допомогою ***MessageBox***. Також він має можливість обрати 7 чи 8 клас за допомогою елемента управління ***ComboBox***.В залежності від класу надалі будуть дещо різнитися завдання, тож створюється змінна public static int klas, у яку ми зберігаємо значення класу.

При натисканні на кнопку «Далі» у папці *Debug* створюється текстовий файл, у який записується ім’я та фамілія учня, а також його клас і у майбутньому туди будуть зберігатися результати проходження завдань.

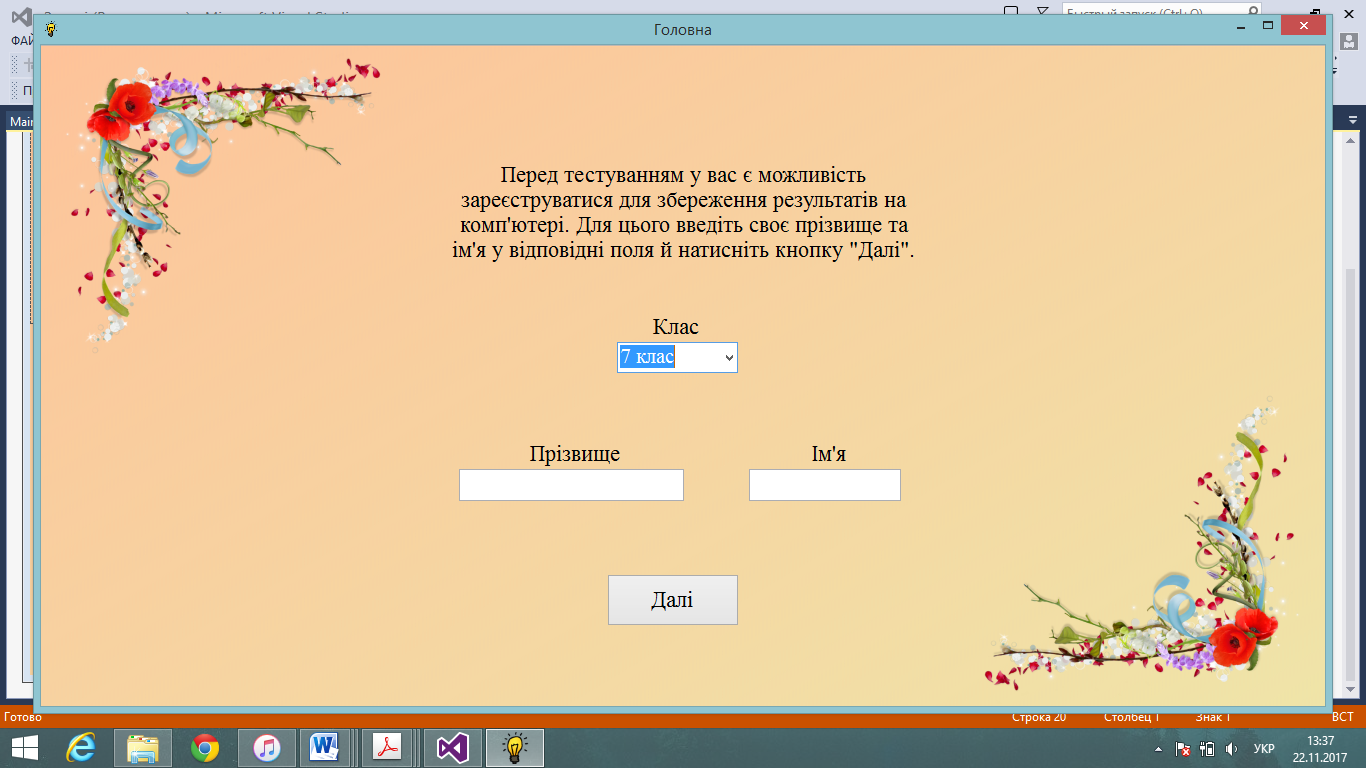


Рисунок 2.Головна

Після цього відкривається форма «Задачі» з хмаринками-***picturebox***, при натисканні на які учень може переходити до вказаних задач.

В залежності від значення змінної *klas,* створеної раніше, на цю форму буде виводиться хмаринка з задачею «Переправа» для 7-х класів або «Переливання» для 8-х.

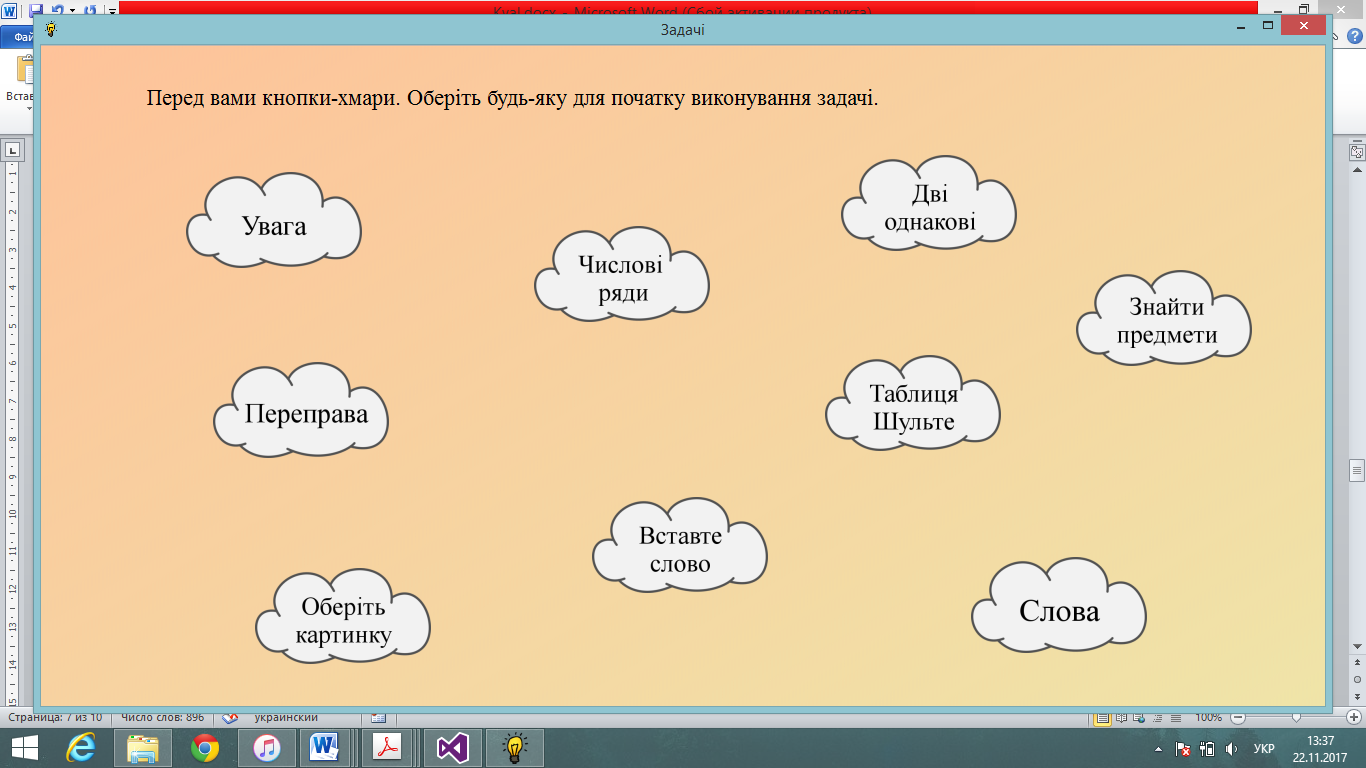


Рисунок 3.Задачі

Для кожної хмаринки вводиться змінна, яка змінює своє значення при натисканні на ***picturebox***. При завантаженні цієї форми перевіряються значення усіх змінних, якщо якась з них змінюється, то відповідний ***picturebox*** змінює значення *Visible* з *true* на *false*. Також перевіряється їх сума, і якщо вона дорівнює 9, то ця форма закривається і замість неї відкривається «Остання сторінка» з результатами.

Далі я хотіла би розглянуть кожну зі своїх задач. Перша з них – «Числові ряди». Перш за все, при завантаженні форми, перевіряється клас. Якщо учень з 8 класу, то йому буде представлено 10 числових рядів, а якщо він з 7 класу, то 3 найважчі числові ряди прибираються, і їх залишається 7. Вони всі зібрані на елементі управління panel. Відповіді потрібно вводити у textbox.

При натисканні на кнопку «Закінчити» вміст усіх ***textbox*** порівнюється з відповідями і якщо вони співпадають, то кількість балів за задачу збільшується на один або декілька балів в залежності від складності побудови ряду.

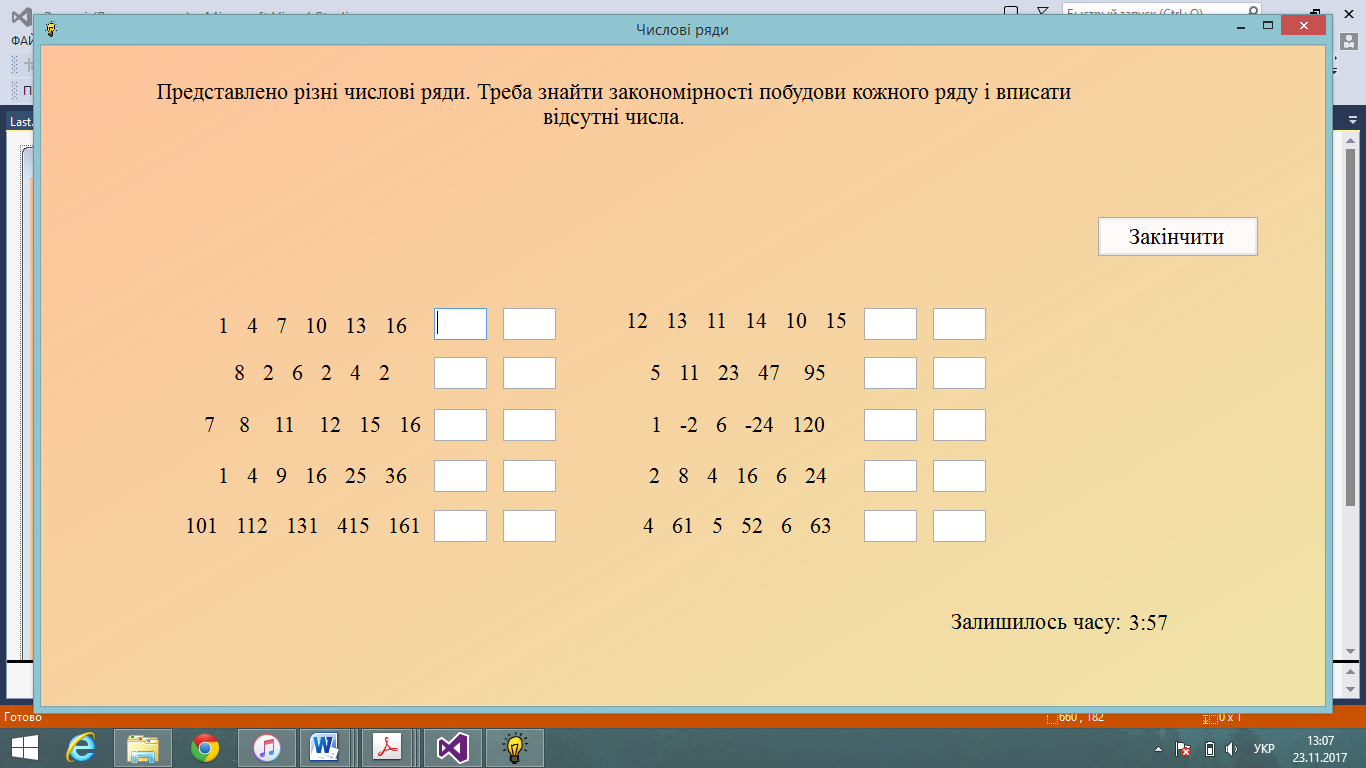


Рисунок 4.Числові ряди

Наступною пропоную розглянути задачу «Дві однакові». У цій задачі застосовується вже зазначений алгоритм створення масиву ***picturebox***:

PictureBox[] pictures = new PictureBox[24];

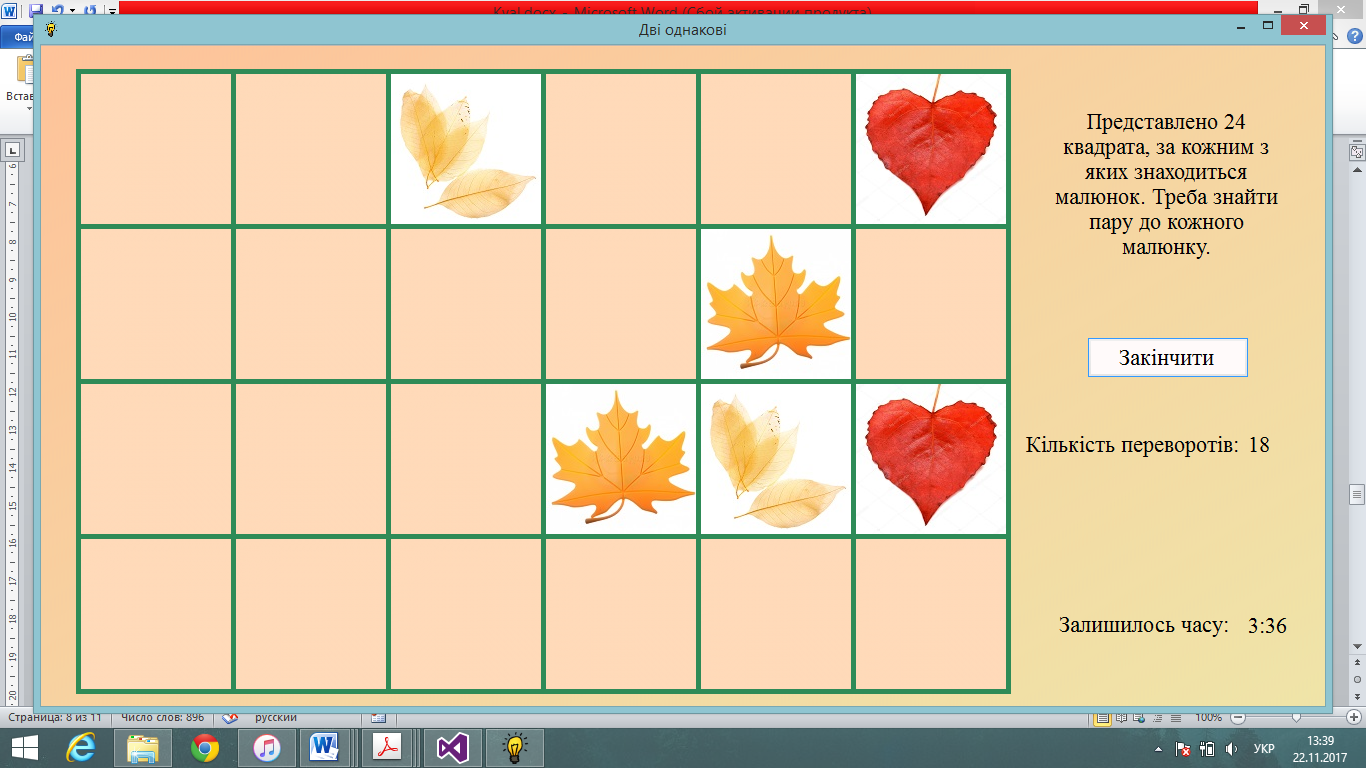


Рисунок 5.Дві однакові

У цій задачі ми використовуємо таку властивість ***picturebox***, як *Tag*, яка нам знадобиться для того, щоб привласнювати картинками цифри, а потім перевіряти, чи вони рівні у однакових картинок. Для зручності усі ***picturebox*** створюються на елементі управління ***panel***.

При запуску задачі усі створені ***picturebox*** «перегорнуті», тобто у кожен з них завантажений малюнок з назвою *pp*. Потім же, залежно від *Tag* обраного квадратика, у нього буде завантажуватися малюнок з папки *Pict*, яка знаходиться у папці *Debug* відповідного номеру.

Спочатку ми створюємо масиви *q* та *w*, які заповнюємо числами від 1 до 12, а потім за допомогою класу *Random* ми «змішуємо» числа у цих масивах:

for (int i = 0; i < 12; i++)

{

x = i + rand.Next(12 - i);

y = q[x];

q[x] = q[i];

q[i] = y;

x1 = i + rand.Next(12 - i);

y1 = w[x1];

w[x1] = w[i];

w[i] = y1;

}

Далі ми задаємо властивість *Tag* елементам масиву ***picturebox*** з номерами від 0 до 11 включно, привласнюючи їй числа з масиву *q*, а з номерами від 12 до 23 – з масиву *w*.

При першому натисканні на будь-який малюнок у змінну *ind1* зберігається значення *Tag* даного ***picturebox***. При другому натисканні та ж сама властивість зберігається у змінну *ind2* і потім перевіряється чи рівні вказані змінні. Якщо це так, то змінна *j* збільшується на один, а коли вона буде дорівнювати 12, то буде запущена функція *Stop*. Також значення змінних *ind1* та *ind2* анулюється. Якщо ж рівність не виконується, то у запускається ще один таймер і через секунду обрані ***picturebox*** знову «перегортаються» і таймер зупиняється.

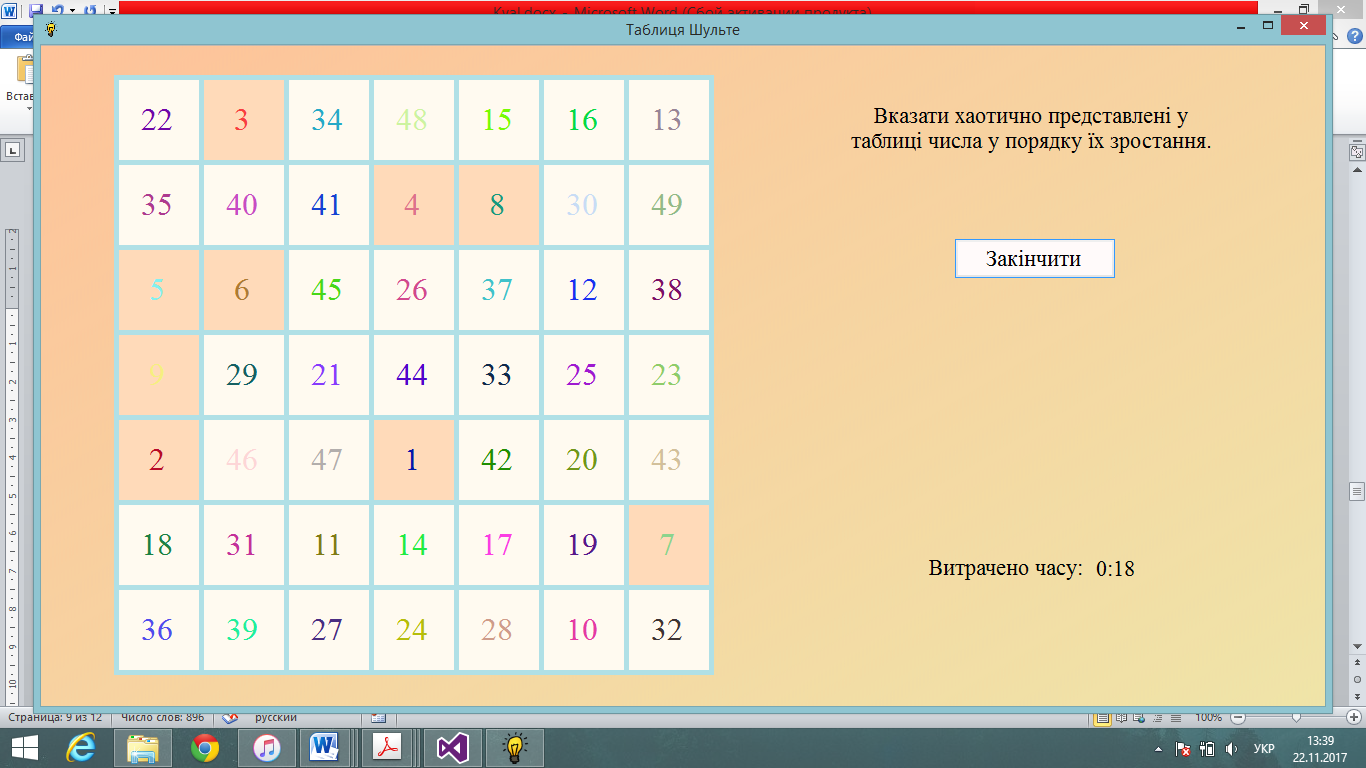


Рисунок 6.Таблиця Шульте

Наступна задача – «Таблиця Шульте». Тут використовується принцип, подібний до того, який був у задачі «Дві однакові».

Ця задача відрізняється від інших тим, що у ній немає оберненого відліку – тут рахується, скільки часу було витрачено на знайдення усіх чисел у порядку їх зростання.

Розмір таблиці залежить від класу, у якому навчається дитина: якщо учень із 7 класу, то таблиця буде розміром 6х6, а якщо з 8, то 7х7. При загрузці форми перевіряється клас і якщо *Main.klas==*7, то *r1=*36 і *r2=*6 *Main.klas==*8, то *r1=*49 і *r2=*7. Змінні потім використовуються при заданні параметрів та у циклах.

При натисканні на кнопку «Почати» створюється масив ***label***, у якому є *r1* елементів (виходить квадрат *r2*х*r2*), який виводиться на ***panel***. Текстом кожного ***label*** є відповідний елемент з масиву *q*, який попередньо був заповнений числами від 1 до *r1* за допомогою класу *Random*. При виведені елементів, вказуючи параметри *Top* и *Left*, використовується двомірний масив.

Тексту кожного елементу масиву за допомогою класу *Random* задається колір:

labels[i].ForeColor = Color.FromArgb(rnd.Next(0, 256), rnd.Next(0, 256), rnd.Next(0, 256));

При заповнюванні елементів, так само, як у попередній задачі, використовується така властивість ***label***, як *Tag*. Туди ми запам’ятовуємо ті самі числа, які йдуть у текст того ж елементу. Потім, при натисканні на будь-який ***label***, у змінну *nom1* (якщо значення змінної *click*, яка відповідає за кількість натискань на будь-які ***label***, дорівнює 1) або *nom2* зберігається значення *Tag* елементу *labels[index]*. Далі перевіряється, чи виконується рівність *nom2 == nom1 + 1*, і якщо це так, то *nom1* привласнюється значення *nom2*, а останнє анулюється. Також, у цьому випадку змінюється забарвлення ***label***. Коли *nom2* дорівнює *r1*, то спрацьовує функція *Stop*.

Наступна задача – «Знайди предмети». У ній масив ***label*** створюється двічі: перший *labels* – на місці знаходження предметів на малюнку, а другий *texts* – для виведення назв предметів, які потрібно буде знайти, адже кожного разу потрібно знаходити різні предмети, які обираються зі списку за допомогою класу *Random*. У першому випадку я всі параметри для кожного елемента задавала власноруч, адже вони залежать від положення предметів на малюнку.

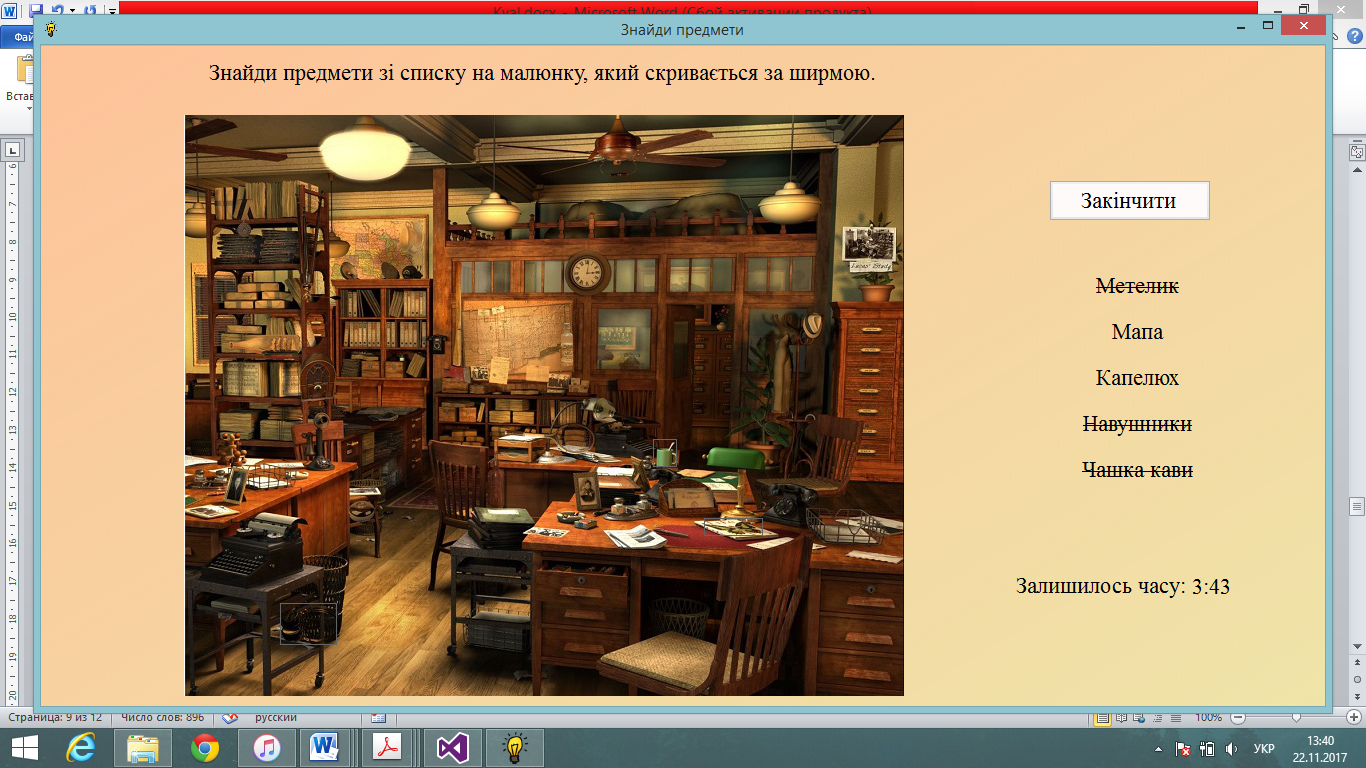


Рисунок 7.Знайди предмети

Малюнок з предметами є частиною фонового зображення, тож, щоб його не було видно з самого початку, на формі знаходиться ***picturebox*** з зображенням ширми. При натисканні на кнопку «Почати» спрацьовує ***timer***, і кожні 20 мс ширина ***picturebox*** зменшується, створюючи враження, що ширма розкривається.

Ще є масив *string*, який я теж задавала власноруч. У ньому зберігається саме список 10 предметів з малюнку, з яких випадково обирається для знаходження лише 5.

С самого початку значення властивості *Enabled* усіх елементів масиву *labels* дорівнює *false*, але при загрузці форми ця властивість тих елементів, які йдуть під номерами значень перших п’яти елементів масиву *q* (який попередньо був «оброблений» функцією *Choose*) змінюється на *true*. Відповідні значення беруться у масиві *texts* і виводяться поруч з малюнком.

Якщо учень знайшов предмет зі списку та натиснув на активний ***label***, то значення *BorderStyle* відповідного елемента змінюється на *FixedSingle*, а потім у циклі перевіряється який саме предмет знайшов учень, порівнюючи значення кожного елементу масиву *texts* з елементом масиву *string* під номером *index* (адже у масиві *labels* і *string* ця властивість відповідно співпадає, тому що вони створювались послідовно). Також змінна *k* збільшується на 1, і коли дорівнює 5, виконується функція *Stop*.

Остання задача – «Слова». Перше, що виконується при її запуску – перевірка класу, у якому навчається дитина. Від цього залежить, який текстовий файл зі словами відкривається:

if (Main.klas == 7)

{

string[] lines = File.ReadAllLines(path1 + "7.txt", Encoding.UTF8);

for (int i = 0; i < 20; i++)

nazva[i] = lines[i];

}

if (Main.klas == 8)

{

string[] lines = File.ReadAllLines(path1 + "8.txt", Encoding.UTF8);

for (int i = 0; i < 20; i++)

nazva[i] = lines[i];

}

Слова, які знаходяться у файлі «7.txt» складаються з 4 слів, а у «8.txt» – з 5 слів.

У цій задачі створюється два масиви ***label***: перший *texts* складається з 7 слів з текстового файлу, а другий *texts1* – з тих же слів, але до них додається ще одне, після чого вони перемішуються. Розглянемо цей процес більш детально.

Спочатку за допомогою функції *Choose()* ми отримуємо масив *q*, який складається з 20 елементів. Потім у циклі задаються параметри кожного елементу, а у властивість *Text* зберігається елемент масиву *string nazva* під номером *q[i]*. Ці ж слова зберігаються у масив *nazva1*.

При натисканні на кнопку «Почати» запускається ***timer3***, і через 10 с ширина ***picturebox1*** збільшується на 100, закриваючи слова. Тут же створюється так само заповнюється масив *texts1*, але на цей раз береться 8 елементів *q[i]*, які також змішуються між собою. Слово, яке додалось зберігається у змінну *odd*. Коли учень натискає на будь-який ***label***, то виконується texts1[index].BringToFront(); та перевіряється, чи дорівнює його текст значенню змінної. Якщо це виконується, то властивість *FontStyle* змінюється на *Strikeout*, а змінна *raz* збільшується на 1. Коли вона дорівнюватиме 3, то виконається функція *Stop*. Якщо ж це не так, то знову виконується функція *Vipoln()*:

void Vipoln()

{

Choose();

for (int i = 0; i < 7; i++)

{

texts[i] = new Label();

texts[i].BackColor = System.Drawing.Color.Transparent;

texts[i].AutoSize = false;

texts[i].TextAlign = label1.TextAlign;

texts[i].Width = 150;

texts[i].Height = 25;

texts[i].Left = left;

texts[i].Top = sh;

sh += 46;

texts[i].Text = nazva[q[i]];

nazva1[i] = nazva[q[i]];

}

nazva1[7] = nazva[q[7]];

odd = nazva1[7];

this.Controls.AddRange(this.texts);

}

Також знову запускається ***timer3***, ширина ***picturebox1*** збільшується на 100 та збільшуються змінні *sh,left* та *sh1*, які відповідають за розташування елементів масивів на формі.

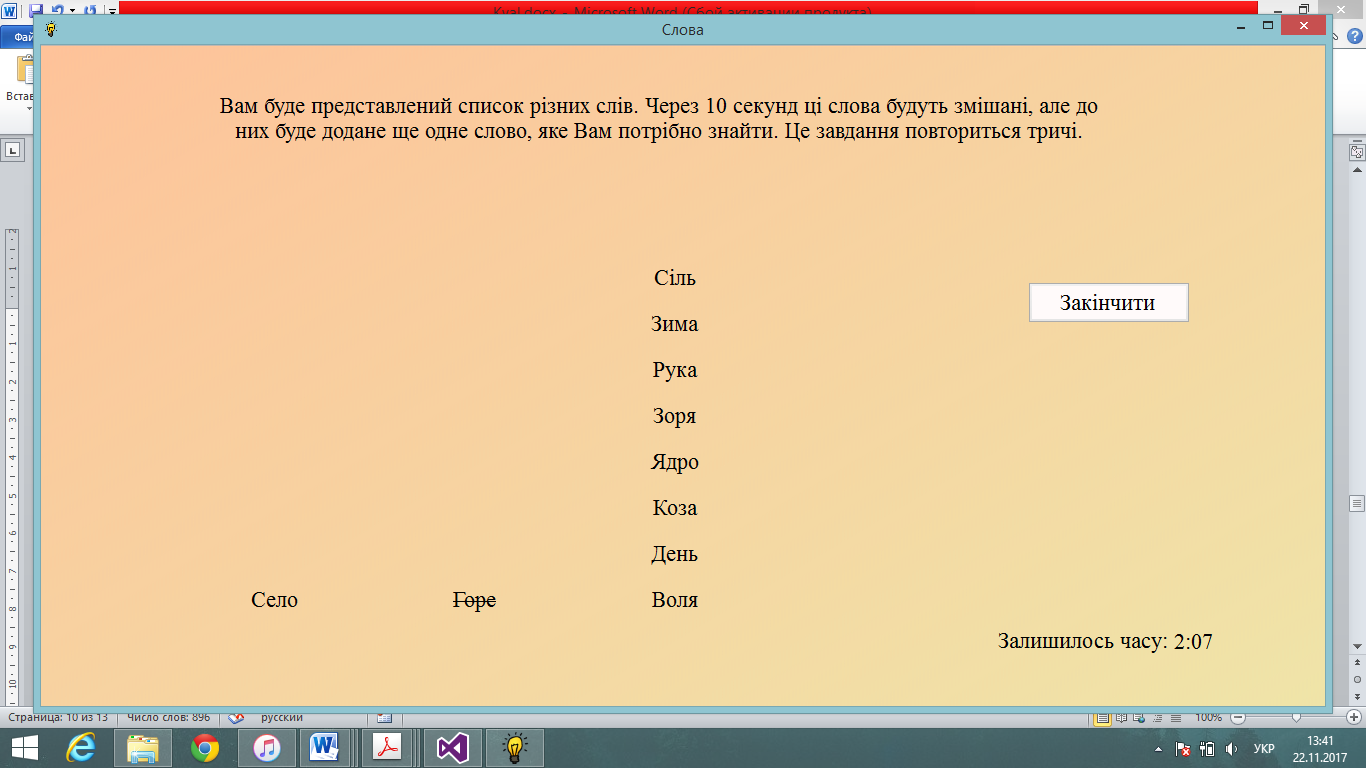


Рисунок 8.Слова

## Структурна схема програми

Результати

Знайдіть предмети

Переправа

Оберіть картинку

Вставте слово

Дві однакові

Таблиця Шульте

Слова

Увага

Переливання

Числові ряди

Головна

Задачі

Титульна сторінка

## Програмні та апаратні вимоги

**Апаратні вимоги:**

1. Процесор частотою 1.6GHz або вище
2. 2048 MB RAM
3. 17,4 MB вільного місця на жорсткому диску
4. Роздільна здатність монітору 1300 x 700 або вище

**Підтримувані операційні системи:**

1. OC Windows 8, Windows 10

## Комплектація програми

* 1. Задачі.exe – файл з практичної частини.
  2. Readme.txt – файл з інформацією, необхідною для прочитання перед запуском програми.
  3. Виноградова.docx – файл з документацією по проекту.
  4. Анотація.docx – файл з анотацією проекту.
  5. Pict – папка з зображеннями, які використовуються в задачі «Дві однакові».
  6. Words – папка з текстовими файлами, які використовуються в задачі «Слова».

## Використані програмні засоби

1. Visual Studio 2013
2. GIMP
3. Microsoft Word
4. Блокнот та Google Chrome

ВИСНОВОК

Результатом нашої роботи є програма – електронна збірка задач для конкурса Турінтел.

Випускна робота на тему «Створення комплексу задач для Турінтелу» зроблена у вигляді програми, що містить десять логічних задач, які перевіряють різні здібності учня, а саме логічні, аналітичні, математичні, лексичні тощо.

Не можна не сказати про те, що головним натхненням у розробці програми виступало те, що вона не зникне, а буде корисною не одному поколінню ліцеїстів та учнів з усієї області.

Більш того, вся зроблена праця допомогла нам поліпшити та поглибити навички програмування мовою C#, а також навчитися користуванню новими елементами управління та функціями.

У ході роботи над програмою мені довелося звернутися до деяких областей інформатики:

1. Програмування: ця частина була найважчою та найцікавішою. Програма була написана мовою C# у Visual Studio 2013.
2. Графіка: деякі зображення редагувалися нами для полегшення програмної частини та підтримки стилю роботи. Графічне редагування здійснювалось у програмах Paint та Gimp 2.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

Інтернет-ресурси:

1. <http://pavlofox.net/tablytsi-shulte/>
2. <http://www.smekalka.pp.ru/series.html>
3. <https://msdn.microsoft.com/ru-ru/library/zkxk2fwf(v=vs.90).aspx>