**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСЬКИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

**КУРСОВА РОБОТА**

з дисципліни

**«Об’єктно-орієнтоване програмування»**

(назва дисципліни)

на тему: **РОЗРОБКА ГРИ «ШИБЕНИЦЯ»**

Студента(ки) 2 курсу 203 - Кб групи

спеціальності «Кібербезпека»

Молібог А. С.

(прізвище та ініціали)

Керівник

Завідуючий кафедри кібербезпеки

(посада, вчене звання, науковий ступінь,

Гордєєв О. О.

прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_

Оцінка: ECTS \_\_\_\_

Члени комісії

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Київ - 2019 рік

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСЬКИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

Спеціальність: «Кібербезпека»

Курс 2 Група 203 - Кб Семестр 1

**ЗАВДАННЯ**

**на курсову роботу студента/студентки**

Молібог Анни Сергіївни

(прізвище, ім’я, по батькові)

1. Тема курсової роботи:

Розробка гри «Шибениця»

2. Термін здачі студентом закінченої роботи 19.12.2019

3. Вихiднi дані до (роботи): інформаційно-аналітичні інтернет джерела, зразки

рішень практичних завдань та задач, нормативно-правова база, щодо оформлен-

ня технічної документації програмних продуктів, початкові значення для об-

робки.

4. Зміст пояснювальної записки (перелік питань, які належить розробити)

Вступ. Формулювання вимог до програми. Проектування та практична

реалізація розв’язку поставленої задачі. Тестування програми і результати

її виконання. Висновки.

5. Перелік графічного матеріалу (з точним зазначенням обов`язкових креслень)

Таблиці та схеми представлено в роботі.

6. Дата видачі завдання “ 10 ” жовтня 2019 р

# Зміст

[ЗМІСТ 3](#_Toc27481919)

[ВСТУП 4](#_Toc27481920)

[РОЗДІЛ 1. ФОРМУЛЮВАННЯ ВИМОГ ДО ПРОГРАМИ 5](#_Toc27481921)

[1.1. Теоретичні відомості 5](#_Toc27481922)

[1.2. Аналіз технічного завдання 6](#_Toc27481923)

[1.3. Обґрунтування алгоритму й структури програми 7](#_Toc27481924)

[РОЗДІЛ 2. ПРАКТИЧНА РЕАЛІЗАЦІЯ РОЗВ’ЯЗКУ ПОСТАВЛЕНИХ ЗАДАЧ 8](#_Toc27481925)

[2.1. Опис програми (обґрунтування вибору мови програмування, опис основних процедур та функцій) 8](#_Toc27481926)

[2.2. Опис змінних та їх ідентифікаторів, що використовуються в програмі 10](#_Toc27481927)

[2.3. Опис алгоритму та коду 12](#_Toc27481928)

[2.4. Опис файлів даних 13](#_Toc27481929)

[РОЗДІЛ 3. ТЕСТУВАННЯ ПРОГРАМИ І РЕЗУЛЬТАТИ ЇЇ ВИКОНАННЯ 14](#_Toc27481930)

[3.1. Опис контрольних прикладів 14](#_Toc27481931)

[3.2. Розробка настанови користувача 17](#_Toc27481932)

[ВИСНОВКИ 20](#_Toc27481933)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 21](#_Toc27481934)

[ДОДАТКИ 22](#_Toc27481935)

# Вступ

Під мовами програмування розуміють штучні мови, призначені для запису програм. Першими мовами програмування були машинні мови. Однак для людини вони дуже незручні, тому в 1950-х рр. було розпочато винайдення мов програмування, які за формою мали бути ближчими до мови математичних формул і людських мов, а також вільними від машинних подробиць. Водночас ці мови мали бути такими, щоб записані ними програми можна було перекласти у відповідні машинні програми за допомогою самої машини. Такі мови невдовзі було розроблено.

Дії комп'ютера в цих мовах мали високий рівень абстракції, тому їх назвали мовами високого рівня [1].

C++ — універсальна мова програмування високого рівня з підтримкою декількох парадигм програмування. Зокрема: об’єктно-орієнтованої та процедурної. Розроблена Б’ярном Страуструпом у 1979 році та названа «С з класами». Страуструп перейменував мову у C++ у 1983 р. Базується на мові Сі.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення [2].

Процес створення програми в найзагальніших рисах вимагає кількох видів діяльності:

– аналіз задачі й уточнення її постановки;

– проектування програми;

– розробка програми (кодування);

– перевірка програми (тестування) [1].

# Розділ 1. Формулювання вимог до програми

## Теоретичні відомості

Комп'ютерна гра – це звичайна програма (впорядкована послідовність системних команд), мета якої забезпечити хороше дозвілля із застосуванням комп'ютера.

Комп'ютерна гра, так само як і звичайна, не є самостійною реальністю, людина що грає усвідомлює нереальність того, що відбувається. Гра обмежується місцем, простором і часом [6].

Перші комп’ютери створювалися в основному для космічної промисловості, а точніше для розрахунку траєкторій польоту ракети. Перед розроблювачами стояло досить складне завдання, тому що потрібно було все розраховувати з урахуванням напрямку вітру, маси ракети й багатьох інших факторів... Розробку програмного забезпечення можна було зрівняти з головоломкою, досить схожою на гру. Комп'ютерні ігри розвивалися із прогресом комп'ютерів, згодом монітор став здатний відбивати не тільки слова, а також графіки, спочатку чорно-білі, потім кольорові. Одночасно збільшувалася швидкодія, розміри оперативної пам'яті, з'явилися мережі.

Першим переломним моментом в історії комп'ютерних ігор варто назвати 1981 рік, саме цього року компанія «IBM» представила свій перший «IBM PC» (персональний комп’ютер). З появою цього ПК комп'ютерні ігри стали розроблятися з величезною швидкістю й до сьогоднішнього часу вдосконалюються.

На сьогоднішній день великий відсоток гостей Інтернет-Кафе складають граючі підлітки , тому що така високотехнологічна розвага як гра не може не залучити до себе. Але з іншої сторони графіка в сучасних комп'ютерах й онлайн іграх це вже не просто дизайн на промисловій основі, а деякий новий напрямок у мистецтві, що може зачаровувати й гіпнотизувати [7].

Заняття комп’ютерними іграми має позитивний вплив на психічні процеси суб’єкта. Користь від комп’ютерної гри полягає у розвитку уваги, моторики, швидкості реакції, сприйняття, розвитку мислення, уяви, креативності.

Комп’ютерні ігри сприяють набуттю спеціальних знань та розвивають когнітивні навички, що входять у загальну структуру діяльності. Ігровий досвід розширює і поліпшує низку умінь і навичок, важливих для навчання, для професійної діяльності та соціальної взаємодії. Віртуальні ігри, в яких моделюються відповідні ситуації, можуть використовуватися для тренування спортсменів, військовослужбовців, хірургів та інших фахівців [8].

Важливою перевагою комп'ютерних ігор є соціалізація людини, захоплюючись кіберспортом, і читаючи, наприклад, новини на спеціалізованому сайті, одиночний гравець може не тільки отримати інформацію про саму гру, але і дізнатися про події, пов'язані з грою, в яких він може взяти участь. Тобто це ще й формування колективу, командного духу, можливість спілкування і отримання нового досвіду в реальності, що також можна віднести до переваг комп'ютерних ігор. Ну і звичайно, не варто забувати про те, що кіберспорт вже давно став тим видом дозвілля, який може принести прибуток, забезпечити отримання призових за якісну гру у складі команди на різного роду змаганнях.

Підкреслимо головні мінуси ігор за комп’ютером, про які не варто забувати. Цілодобове сидіння перед монітором – стовідсоткова шкода здоров’ю. Треба робити перерву, причому достатньо велику, змінити вид діяльності, наприклад, піти на вулицю [9].

## Аналіз технічного завдання

Необхідно розробити алгоритм та написати програму для гри «Шибениця». Програма має загадати слово. Мета користувача ‒ відгадати це слово, раніше ніж буде намальовано малюнок. Програма має обмежувати кількість невдалих спроб.

Додаток працює англійською мовою у консольному режимі. Користувачеві буде відома лише кількість літер у загаданому слові. За розвитком подій, програма буде інформувати користувача, які літери він уже використав.

Правила гри:

Користувач обирає будь-яку літеру англійського алфавіту. Якщо у загаданому комп’ютером слові ця літера є, то всі аналогічні букви у слові будуть відкриті та розставлені на свої місця. Якщо ж такої літери у слові немає, то на рисунку малюється частина людини на шибениці.

Користувач виграє, якщо слово буде відгадано раніше, ніж рисунок буде повністю намальованим. Якщо рисунок буде намальованим раніше ніж буде відгадано слово, гра вважатиметься програною.

## Обґрунтування алгоритму й структури програми

Після запуску програми, користувач вводить з клавіатури по одній літері. Далі програма перевіряє, якщо у загаданому слові є введена користувачем літера, то вона вписується на відповідне місце або місця у слові, в залежності від того, скільки разів дана літера зустрічається в загаданому програмою слові. Якщо ж введена літера не зустрічається у слові, то у відповідному блоці малюється частина рисунку; у слово літера не вписується (блок-схему програми див. у додатку А).

Для коректної роботи, програма рахує кількість помилок. Кожній помилці, за порядковим номером, відповідає конкретна частина малюнку. Загалом у користувача є можливість зробити десять помилок. У разі допущення десяти помилок, програма виводить на екран повідомлення про завершення гри «GAME OVER». Якщо ж користувач вгадає слово, до того як рисунок буде повністю намальовано, то програма виведе повідомлення про перемогу користувача: «YOU WON!».

# Розділ 2. Практична реалізація розв’язку поставлених задач

## Опис програми (обґрунтування вибору мови програмування, опис основних процедур та функцій)

Для написання програми використовується С++. Це універсальна мова програмування, так як компілятори С++ є у кожній операційній системі. Також перевагою є висока швидкість виконання коду. Будь-який тест з продуктивності мов програмування покаже, що С++ стоїть на лідируючому місці. Java, JavaScript, C#, як і безліч інших популярних мов програмування, містять в основі принципи C++. Тому учбові програми багатьох ВНЗ містять курс С++ [3].

Створюється програма, яка випадковим чином обирає слово і завдання користувача вгадати його. Користувач вводить по одній літері з клавіатури. За кожний крок програми відповідає відповідна функція.

Функція — це іменована логічно завершена сукупність оголошень і операторів, призначених для виконання певної задачі.

Усі функції програми, повинні бути оголошені та визначені до їх першого використання. Оголошення функції задає ім’я функції, тип значення, що повертає функція (якщо воно є), а також імена та типи аргументів, які можуть передаватися як у функцію, так і з неї. Визначення функції — це задання способу виконання операцій.

Слід нагадати, що серед функцій програми повинна бути одна з ім’ям main (головна функція), яка може знаходитися в будь-якому місці програми. Ця функція виконується завжди першою і закінчується останньою [4].

У програмі є 7 функцій (не рахуючи головної функції main()). Далі наведено список та опис цих функцій.

Функція PrintMessage(string message, bool printtop = true, bool printbottom = true) відповідає за вивід та розміщення по центру інформації на екран у вигляді панелей. String message – інформація що міститься у панелі; bool printtop – верхня частина панелі; bool printbottom – нижня частина панелі.

Функція Rules(string rule) передає на екран правила гри. Змінна string rule містить у собі строку з правилами.

Функція Draw(int guess=0) малює частини шибеника відповідно до кількості помилок (значення змінної guess).

Функції PrintLetters(string input, char from, char to) та PrintAvailableLetters(string taken) пов’язані між собою. Перша функція перевіряє введену користувачем літеру string input, якщо вона знаходиться у проміжку [char from, char to], то вона виводиться на екран, інакше – не виводиться. Таким чином, функція виводить на екран невикористані літери, та приховує вже введені користувачем літери у програму. Друга ж функція виводить назву панелі з доступними літерами та, власне доступні літери розбиваючи їх на два рядки: перший – від А до М, другий – від N до Z.

Функція PrintWordAndCheckWin(string word, string guessed) друкує вгадані користувачем літери на відповідні їм місця у слові, невідгадані літери позначаються символом “\_”. Також функція перевіряє чи вгадав користувач слово. Є змінна won типу bool, якій надається значення true. Далі у циклі перевіряється чи є введена з клавіатури літера у загаданому програмою слові, якщо ні, то змінній won надається значення false. Так перевіряється кожна введена буква з кожною буквою у загаданому слові. Якщо були введені усі літери загаданого слова, то змінна won знову приймає значення true.

Функція LoadRandomWord(string path) завантажує слова з файлу (додаток Б) у динамічний масив, випадковим чином обирає одне з них та присвоює його змінній word. Функція повертає значення змінної word.

Функція TriesLeft(string word, string guessed) підраховує кількість помилок error. У циклі перевіряється чи є введена з клавіатури літера guessed у загаданому програмою слові word, якщо ні, то змінна error інкрементується.

## Опис змінних та їх ідентифікаторів, що використовуються в програмі

Кожна програма потребує виконання різноманітних обчислень, для здійснення яких використовуються вирази, що складаються з операндів, знаків операцій і дужок. Операнди задають дані для обчислень, а операції задають дії, які необхідно виконати над цими даними. Операнд є, у свою чергу, виразом, що в окремому випадку може бути константою або змінною.

Змінна — це іменована область пам’яті, у якій зберігаються дані визначеного типу. Основні типи даних часто називають арифметичними, тому що їх можна використовувати в арифметичних операціях. Для опису основних типів мови C++ використовують такі службові слова:

* int (цілий);
* char (символьний);
* bool (логічний);
* float (дійсний);
* double (дійсний з подвійною точністю);

Змінна має ім’я, розмір та інші атрибути, такі як видимість, час існування тощо. Ім’я змінної служить для звертання до області пам’яті, у якій зберігається її значення. Перед використанням будь-яка змінна повинна бути описана, при цьому для неї резервується деяка область пам’яті, розмір якої залежить від конкретного типу змінної. Під час виконання програми змінна може приймати рiзнi значення [5].

Змінні у програмі можуть порівнюватися. Для цього, використовуються вбудовані знаки порівняння (табл. 2.1.). Для їх використання непотрібно підключати додаткову бібліотеку.

Таблиця 2.1.

Знаки порівняння в С++

|  |  |  |
| --- | --- | --- |
| **Порівняння у С++** | **Опис** | **Приклад** |
| > | більше ніж | x>0 |

Продовження таблиці 2.1.

|  |  |  |
| --- | --- | --- |
| < | менше ніж | y<z |
| >= | більше або дорівнює | y>=x |
| <= | менше або дорівнює | z<=8.56 |
| == | перевірка на рівність | x==0.7 |
| != | не дорівнює | x!=y |
| && | логічне І | x>0 && x<1 (подвійна нерівність 0<x<1) |
| | | | логічне АБО | z<8 | | z>10 |

Для кожної функції програми є свої змінні, далі іде перелік та опис цих змінних.

*Змінні для головної функції int main():*

guesses – рядок використаних літер;

wordToGuess – загадане слово (приймає значення функції LoadRandomWord);

tries = 0 – кількість допущених помилок (приймає значення функції TriesLeft);

win = false – перевірочна змінна, яка показує вгадав користувач слово чи ні (набуває значення функції PrintWordAndCheckWin).

*Змінні для функції PrintMessage:*

message – інформація, що розміщується у центрі панелі;

printtop = true – змінна для створення верхньої частини панелі;

printbottom = true – змінна для створення нижньої частини панелі;

front = true – допоміжна змінна для центрування тексту.

*Змінні для функції Rules:*

rule – текст правила;

front = true – допоміжна змінна для центрування тексту.

*Змінні для функції Draw:*

guess=0 кількість помилок.

*Змінні для функції PrintLetters:*

input – введені користувачем літери;

from – початок буквеного проміжку;

to – кінець бук венного проміжку;

s – рядок з введеними літерами.

*Змінні для функції PrintAvailableLetters:*

taken – введені літери.

*Змінні для функції PrintWordAndCheckWin:*

word – загадане слово;

guessed – введена літера;

won = true – перевірочна змінна;

s – рядок із загаданим словом.

*Змінні для функції LoadRandomWord:*

path – посилання на файл;

word – загадане слово;

v – динамічний масив рядків;

reader – звертання до файлу;

randomline – випадкове слово у динамічному масиві рядків.

*Змінні для функції TriesLeft:*

word – загадане слово;

guessed – введена літера;

error – кількість помилок.

## Опис алгоритму та коду

Після запуску програми, на екран виводиться інтерфейс, що розроблено у вигляді панелей (код програми у додатку Б). До того, як гравець почне вводити по черзі літери, програма, використовуючи функції обирає випадкове слово з відповідного файлу. Замість кожної літери загаданого слова, на екран виведено символ “\_” (символи розділені пробілами). За допомогою цього, користувач може визначити скільки літер міститься у загаданому слові.

Після ознайомлення з інтерфейсом, правилами гри та програми, користувач вводить з клавіатури будь-яку літеру. Функції програми перевіряють її. Якщо ця літера є у загаданому слові, то вона викреслюється із списку доступних літер та вписується у панелі з загаданим словом на відповідне їй місце або місця, якщо дана літера зустрічається у слові більше одного разу. У випадку, коли введена літера не зустрічається у загаданому слові, вона видаляється із панелі з доступними літерами, а у відповідній панелі, функція Draw малює частину шибеника, відповідно до кількості помилок. На панелі з загаданим словом літера не з’являється.

У користувача є можливість допустити десять помилок. Як тільки користувач введе останню правильну літеру, при цьому припустивши не більше десяти помилок, загадане слово буде повністю відображено на відповідній панелі і з’явиться нова панель з повідомленням про виграш (YOU WON!). Якщо ж користувачу не вдасться відгадати слово не припустивши десяти помилок, то з’явиться інша панель, з повідомленням про програш (GAME OVER).

Зауваження: до програми можна вводити тільки великі літери англійського алфавіту. Введення цифр, будь-яких символів, малих літер англійського алфавіту або літер (малих та великих) будь-якого іншого алфавіту, вважатиметься за помилку і призведе до виведення на екран наступної частини рисунку шибеницю.

## Опис файлів даних

Випадковий вибір слова у програмі виконується функцією LoadRandomWord. Вона передає значення рядків з текстового файлу words.txt (зміст файлу у додатку В) у динамічний масив рядків, з якого вже за допомогою функції rand() випадково вибирає рядок із словом, та передає його значення змінній word.

Текстовий файл містить 20 слів, записаних великими літерами англійською мовою. Кожне слово записане з нового рядка. Слова не мають містити у собі пробілів або знаків дефісу «-» та тире «‒».

Файл можна змінювати: додавати та видаляти слова. Відповідно у програми буде більший або менший вибір слів. Це можливо завдяки тому, що масив об’явлено динамічно і для функції rand() кінцевою границею генерування випадкового рядка, є розмір масиву, а отже останній рядок текстового файлу.

# Розділ 3. Тестування програми і результати її виконання

## Опис контрольних прикладів

Розглянемо приклад. Запускається програма. Виводиться початковий екран у вигляді панелей (рис. 3.1.). Інформаційна панель ознайомлює користувача з правилами гри. На панелі з загаданим словом п’ять символів “\_”, це означає, що у слові п’ять літер. Велика пуста панель відведена під малювання шибениці.

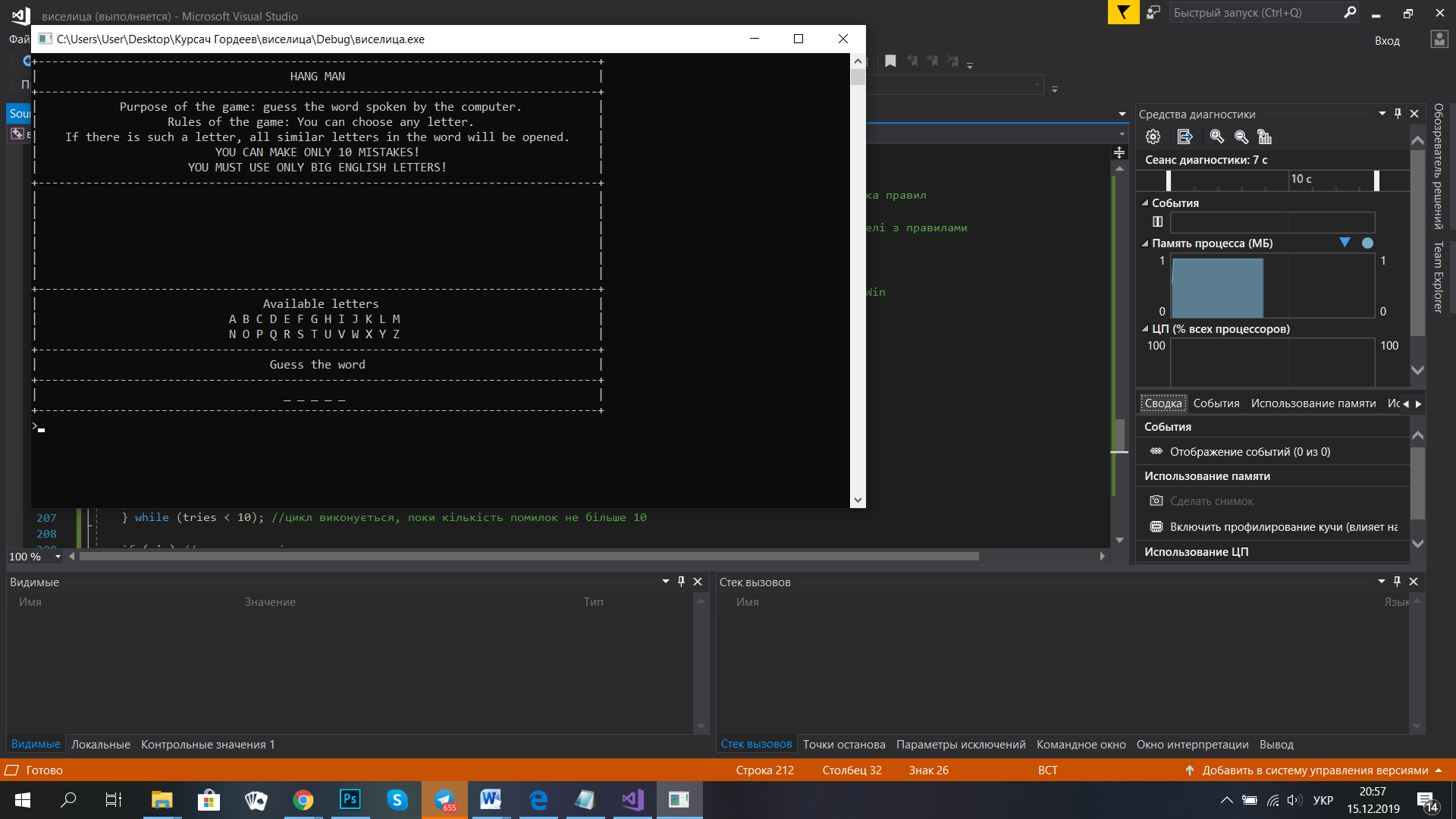


Рис. 3.1. Початковий екран програми

Першим кроком, користувач вводить літеру «О». Ця літера є у загаданому слові, тому вона вписується на відповідне місце у загаданому слові та видаляється з панелі доступних літер (рис. 3.2).

Користувачем було введено ще 2 літери: «Н» та «М» (рис. 3.3.). Літера «Н», так само, як і літера «О», є у загаданому програмою слові тому вона також вписується на своє місце у загаданому слові та видаляється з панелі доступних літер.

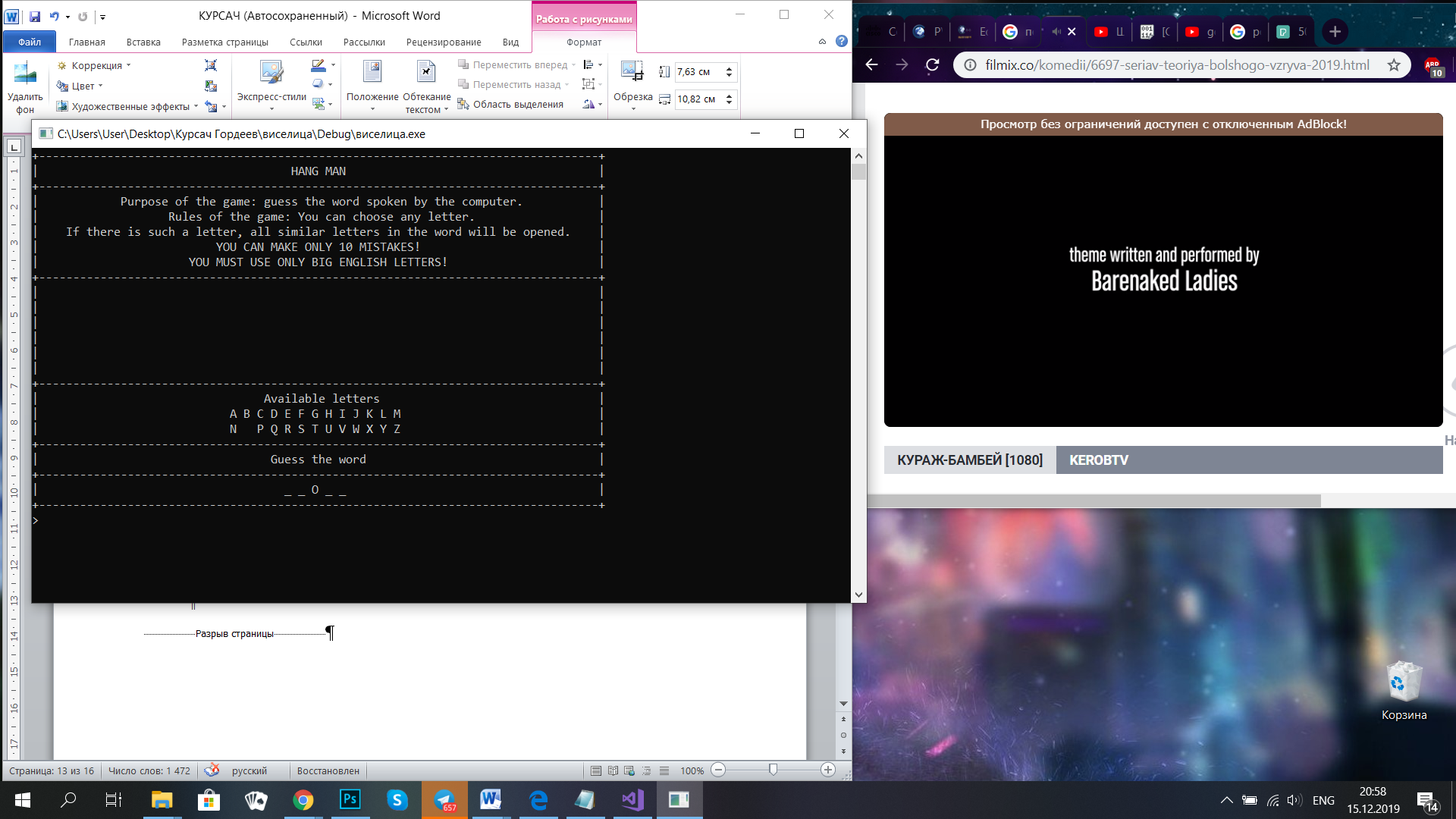


Рис. 3.2. Введення першої літери

Як видно з виведеної на екран інформації, літера «М» не з’явилася на панелі з загаданим словом, проте, вона видалилася з панелі доступних літер, що свідчить про те, що вона була введена. Це свідчить про те, що літери «М» у слові немає і користувач припустився помилки. Як наслідок, на панелі для малюнку з’явилася перша частинка шибениці.

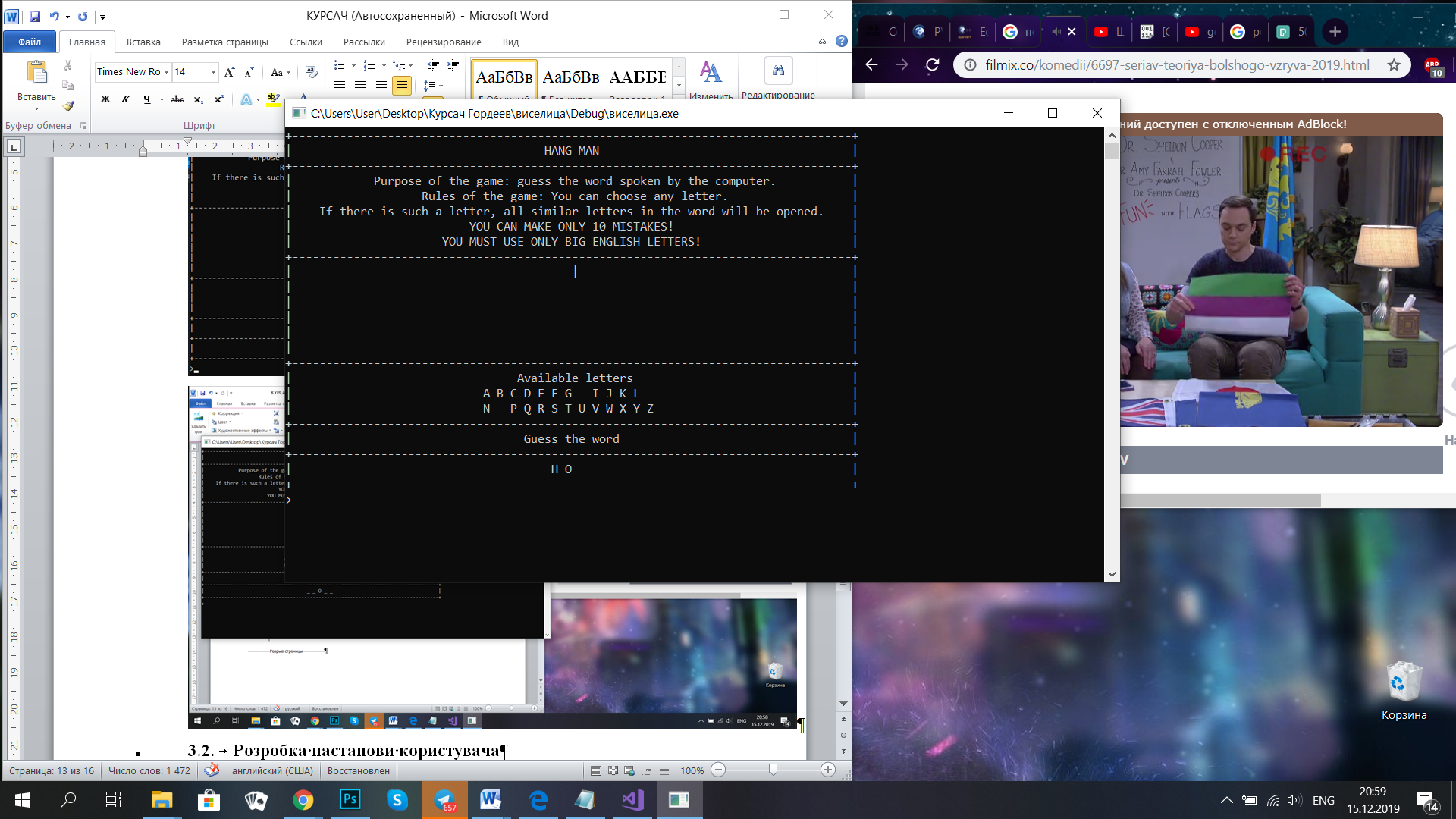


Рис. 3.3. Введення наступних двох літер

Користувач ввів ще чотири літери: «P», «N», «E», «T». Літери «P», «N», «E» містяться у загаданому слові. Літера «T» ‒ помилкова. Усі введені літери видалилися з панелі доступних літер. Відгадані літери вписуються програмою на відповідні їм місця у слові. За помилкову літеру «T», у відповідній панелі було домальовано одну частину рисунку шибениці.

Слово відгадано (загадане слово ‒ «PHONE»), про що свідчить нова панель з повідомленням «YOU WON!» (рис. 3.4.).

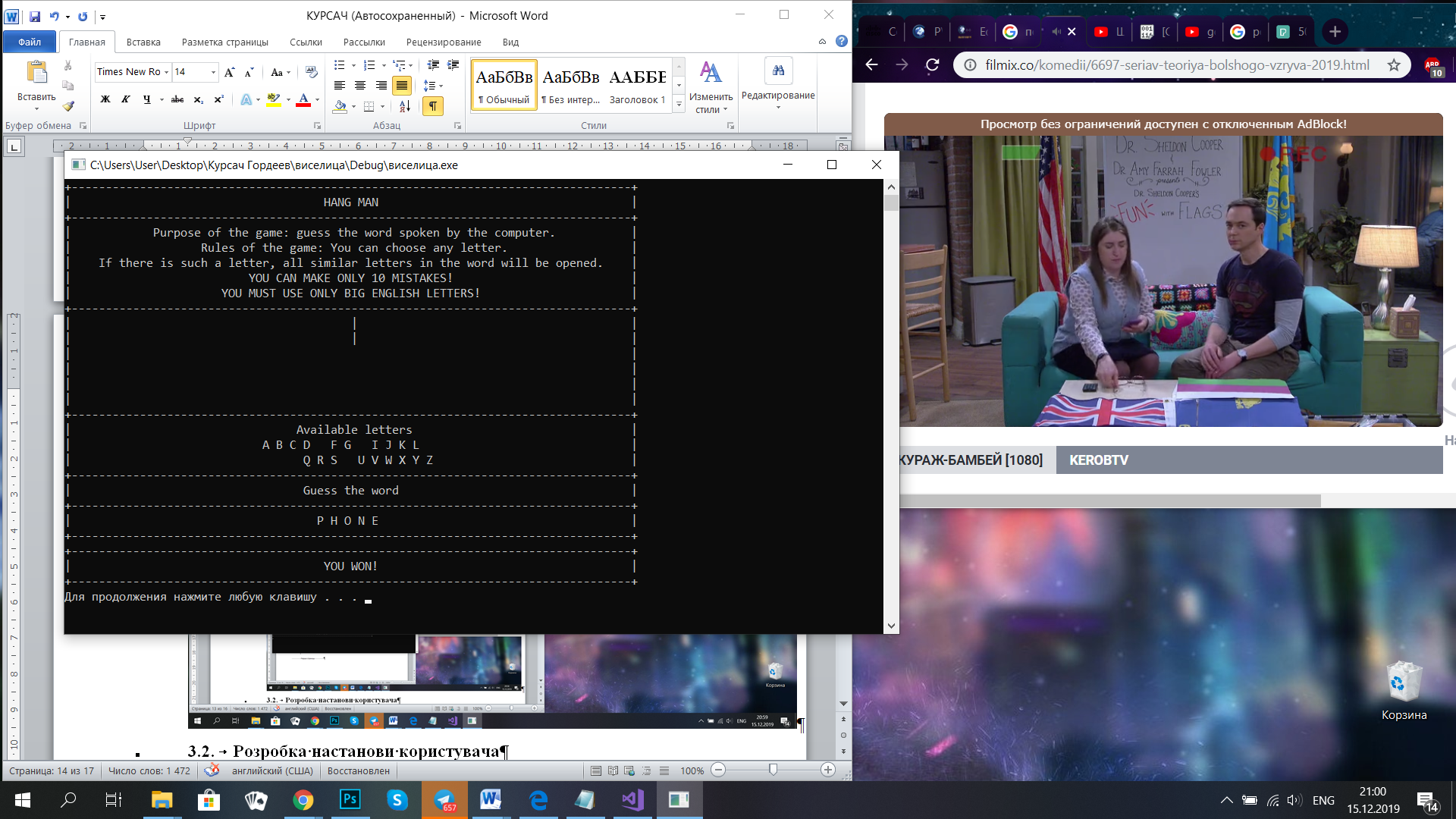


Рис. 3.4. Слово відгадано

Усі введені літери видалені з панелі доступних літер, загадане слово повність відкрито. Як видно з рисунка 3.4., на екран виведено лише 2 частини малюнка, що свідчить про те, що було допущено лише дві помилки.

Розглянемо приклад, коли користувач не відгадав слово (рис. 3.5.). Було загадано слово з одинадцяти літер. Користувач відгадав три літери, одна повторювалась у слові два рази, тому відкрито чотири букви. Проте, користувач також ввів 10 неправильних літер, тому малюнок намальовано повністю. Панель з повідомленням про програш з’являється нижче місця для вводу літери. Усі введені літери прибрані з панелі невикористаних літер.

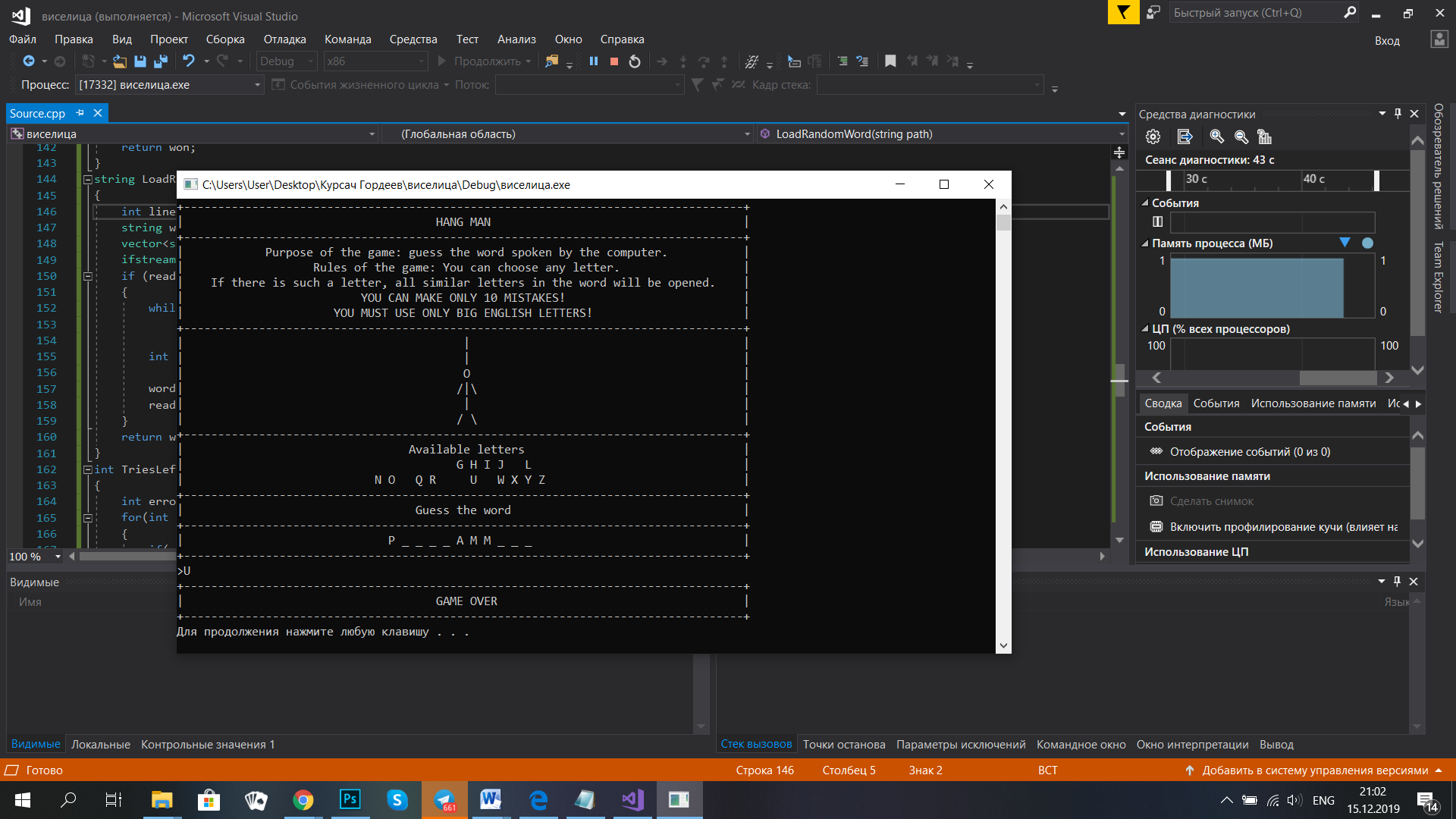


Рис. 3.5. Випадок програшу

## Розробка настанови користувача

На початковому екрані програми розміщено сім панелей (рис. 3.6.). З них чотири неактивні та три взаємодіють з користувачем. Перша панель містить назву гри «HANG MAN», друга панель містить інформацію про ціль гри, правила, а також особливі умови поставлені саме цією програмою (умови виведені на екран великими англійськими літерами). Наступна панель під час запуску порожня. У ній малюються частини шибениці, коли користувач допускає помилку. Четверта панель є назвою п’ятої панелі, що містить доступні літери «Available letters». Після введення користувачем літери у програму, відповідна літера прибирається з даної панелі. Таким чином користувач бачить, які літери він ще не використовував, що зменшить вірогідність введення однієї і тієї самої літери декілька разів. Наступна панель є заголовком до панелі з загаданим словом «Guess the word». На рисунку 3.6. панель повністю заповнена символами “\_”, так як це лише приклад, у діючій програмі кількість символів відповідає кількості літер у загаданому слові.

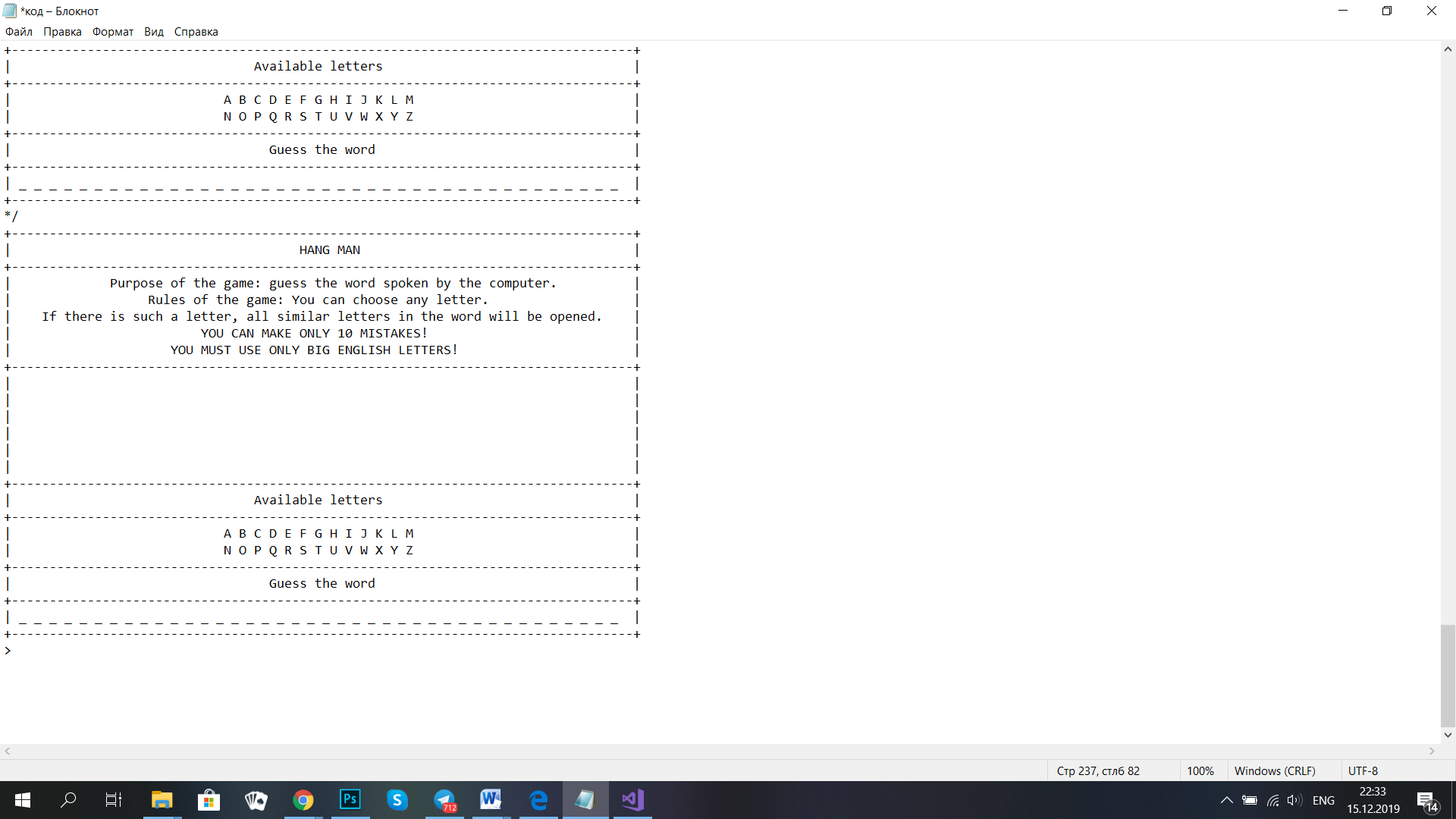


Рис. 3.6. Інтерфейс програми

Місце для введення літер знаходиться під останньою панеллю та умовно позначається символом “>” (рис. 3.7.). Введення літери починається одразу після нього. Літера обробляється програмою тільки після натискання клавіші «Enter».

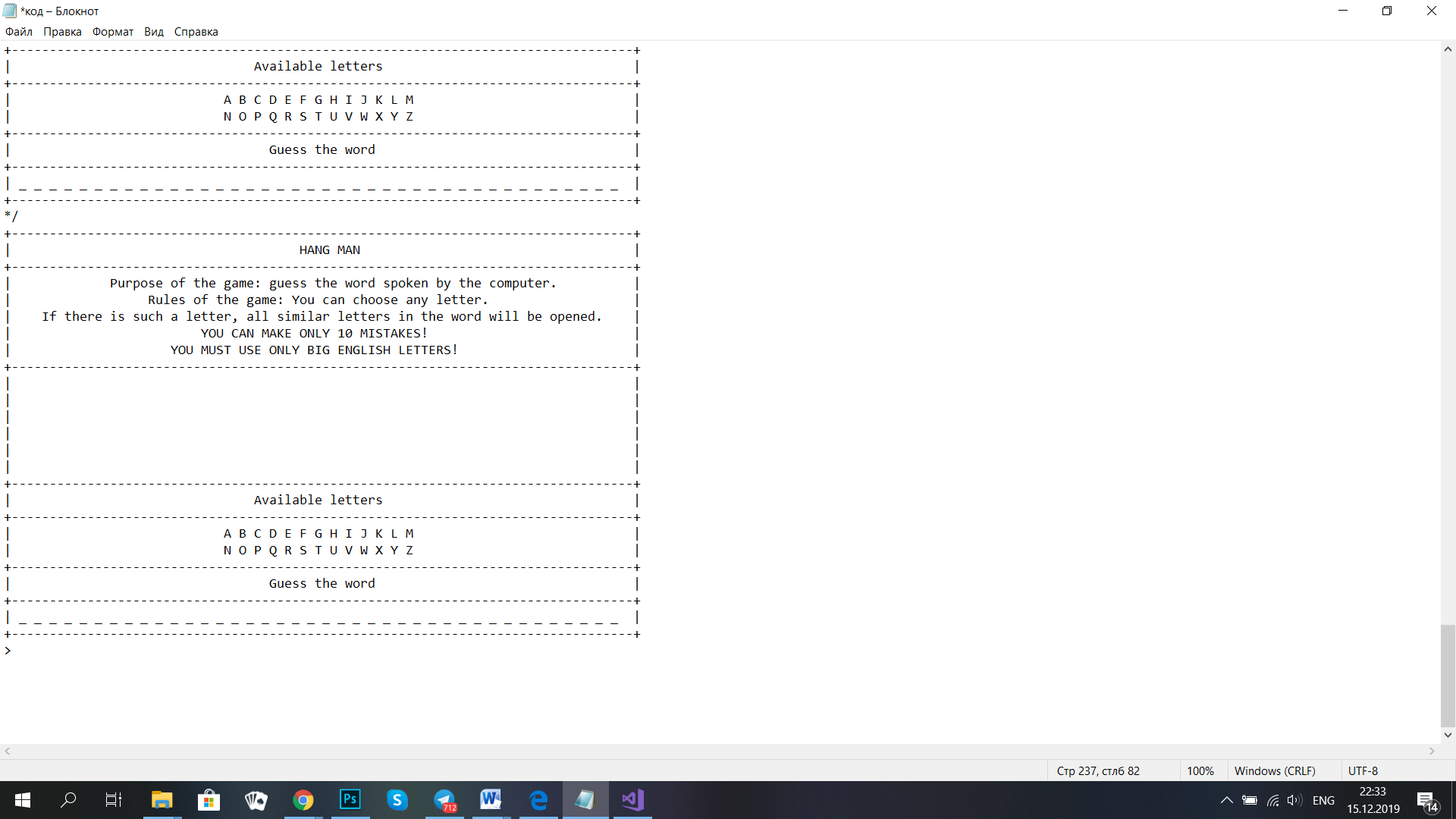


Рис. 3.7. Умовний символ для введення літер

Після того, як користувач відгадує слово, виводиться панель з повідомленням про виграш (рис. 3.8.).

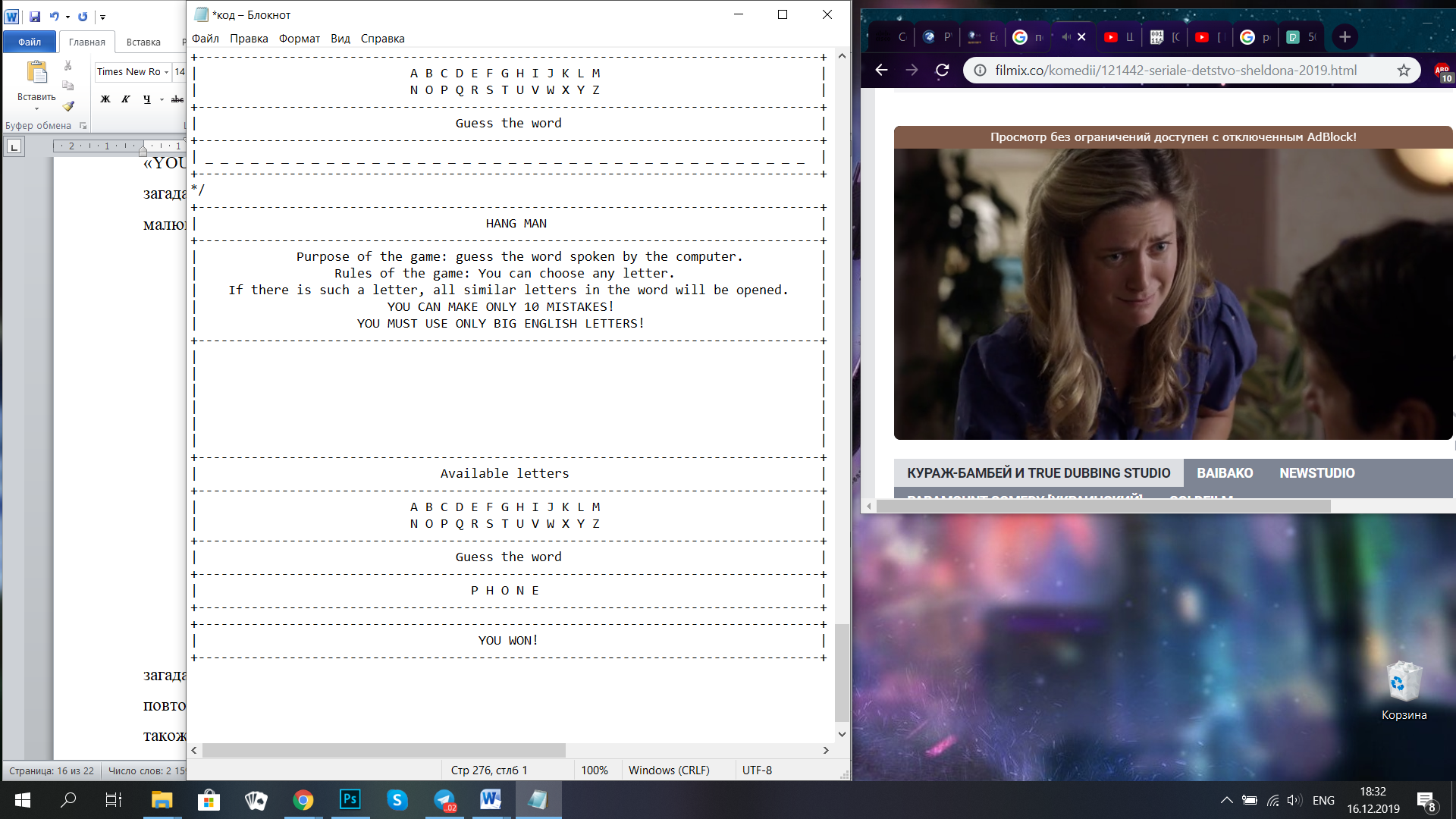


Рис. 3.8. Панель з повідомленням про виграш

Коли користувач програє, з’являється нова панель з повідомленням про програш (рис. 3.9.).

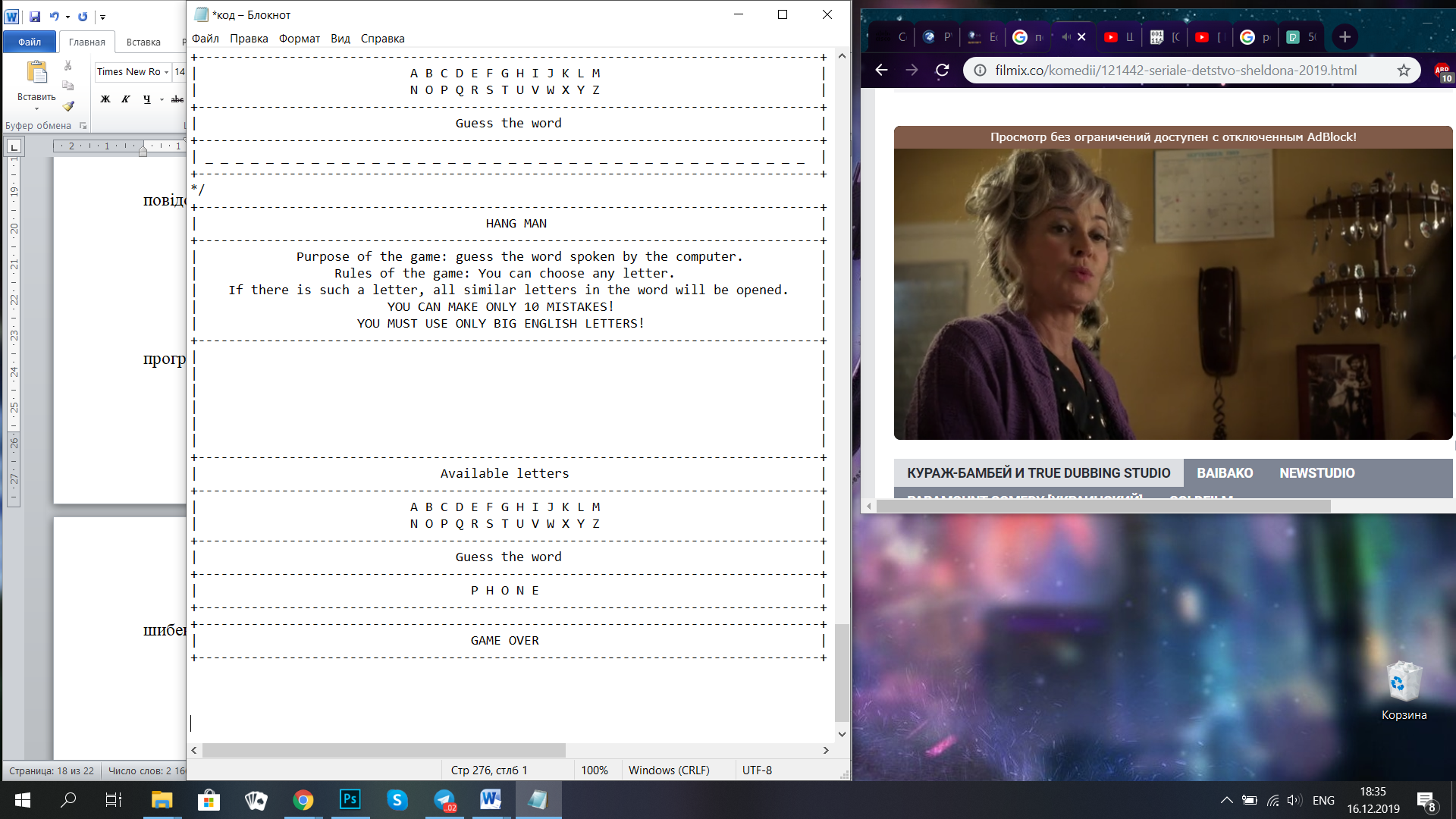


Рис. 3.9. Панель з повідомленням про програш

У панелі, що була порожньою під час запуску, буде намальовано шибеницю так, як було допущено десять помилок (рис. 3.10.).

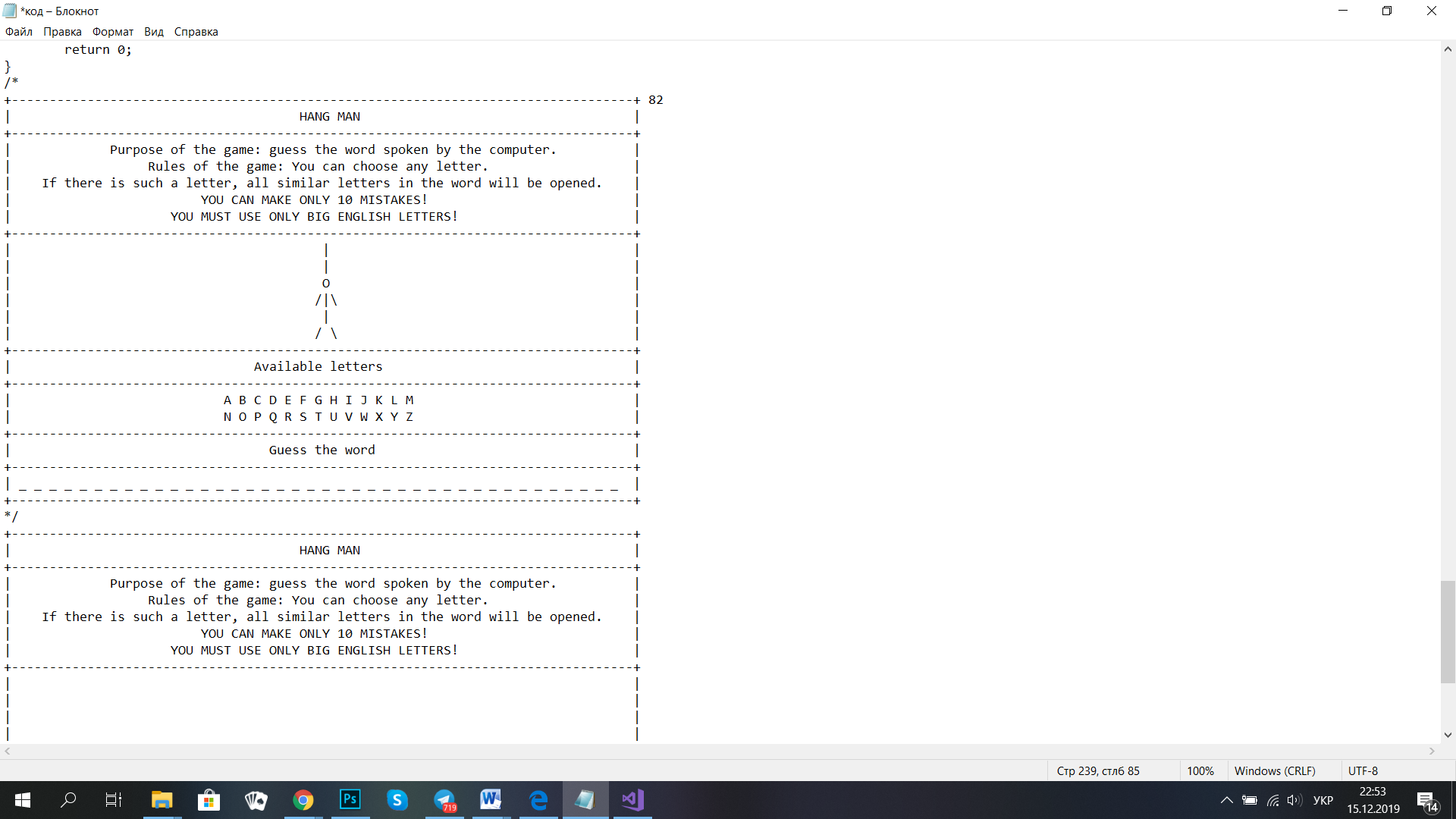


Рис. 3.10. Повністю намальована шибениця – ознака програшу

Програма працює з особливими умовами для вводу даних. Введення можливе лише великими літерами англійського алфавіту. Якщо буде введено малі літери, або літери (як малі, так і великі) інших алфавітів, або символи чи цифри, дані будуть враховані за помилку і буде домальовано частину малюнка.

# Висновки

У даній курсовій роботі була розроблена гра «Шибениця». Для досягнення поставленої мети були розглянуті наступні теми:

* функції;
* робота з рядками;
* булєві змінні;
* логічні функції.

Програма працює у консольному режимі. Розроблений доступний для розуміння користувача інтерфейс. Проте у програми є свої недоліки:

* не реалізована перевірка вхідних даних;
* не реалізована можливість обирати мову гри.

Зазначені зауваження не є принциповими та не впливають на коректність роботи програми. Тому, можна вважати, що всі поставлені задачі виконані.

Під час використання цієї програми, можна гарно провести час та розвивати логічне мислення.

Були отримані практичні навички з написання кодів мовою програмування С++.

# Список використаних джерел

1. Вступ до програмування мовою С++. Організація обчислень : навч. посіб. / Ю. А. Бєлов, Т. О. Карнаух, Ю. В. Коваль, А. Б. Ставровський. – К. : Видавничо-поліграфічний центр "Київський університет", 2012. – 175 с. с.: іл. ISBN (укр.);
2. Огляд і основи мови програмування C++ [Електронний ресурс] // Портал знань. – 2018. – Режим доступу до ресурсу: http://www.znannya.org/?view=Cpp\_basics;
3. Почему C++ крут, актуален и бессмертен [Електронний ресурс] // VC.ru. – 2018. – Режим доступу до ресурсу: https://vc.ru/hr/50161-pochemu-c-krut-aktualen-i-bessmerten;
4. Поняття функції [Електронний ресурс] // С++ програмування. – 2019. – Режим доступу до ресурсу: http://cpp.dp.ua/ponyattya-funktsiyi/;
5. Змінні [Електронний ресурс] // С++ програмування. – 2019. – Режим доступу до ресурсу: http://cpp.dp.ua/zminni/;
6. Значення терміну «комп’ютерна гра», класифікація комп’ютерних ігор [Електронний ресурс] // StudFiles. – 2015. – Режим доступу до ресурсу: https://studfile.net/preview/4364269/page:2/;
7. Історія створення комп’ютерних ігор, та їх розвиток на сучасному етапі [Електронний ресурс] // StudFiles. – 2015. – Режим доступу до ресурсу: https://studfile.net/preview/4364269/page:3/;
8. Гордієнко А. В. Комп’ютерні ігри та їхні позитивні психологічні ефе [Електронний ресурс] / А. В. Гордієнко // НАУКОВІ ЗАПИСКИ НаУКМА. – 2017. – Режим доступу до ресурсу: http://ekmair.ukma.edu.ua/bitstream/handle/123456789/12402/Hordiienko\_Kompiuterni\_ihry.pdf;
9. Комп’ютерна гра: користь чи навпаки? [Електронний ресурс] // 0629.com.ua. – 2016. – Режим доступу до ресурсу: https://www.0629.com.ua/news/1049864/komputerna-gra-korist-ci-navpaki;

# Додатки

**Додаток А.1. Блок-схема програми**

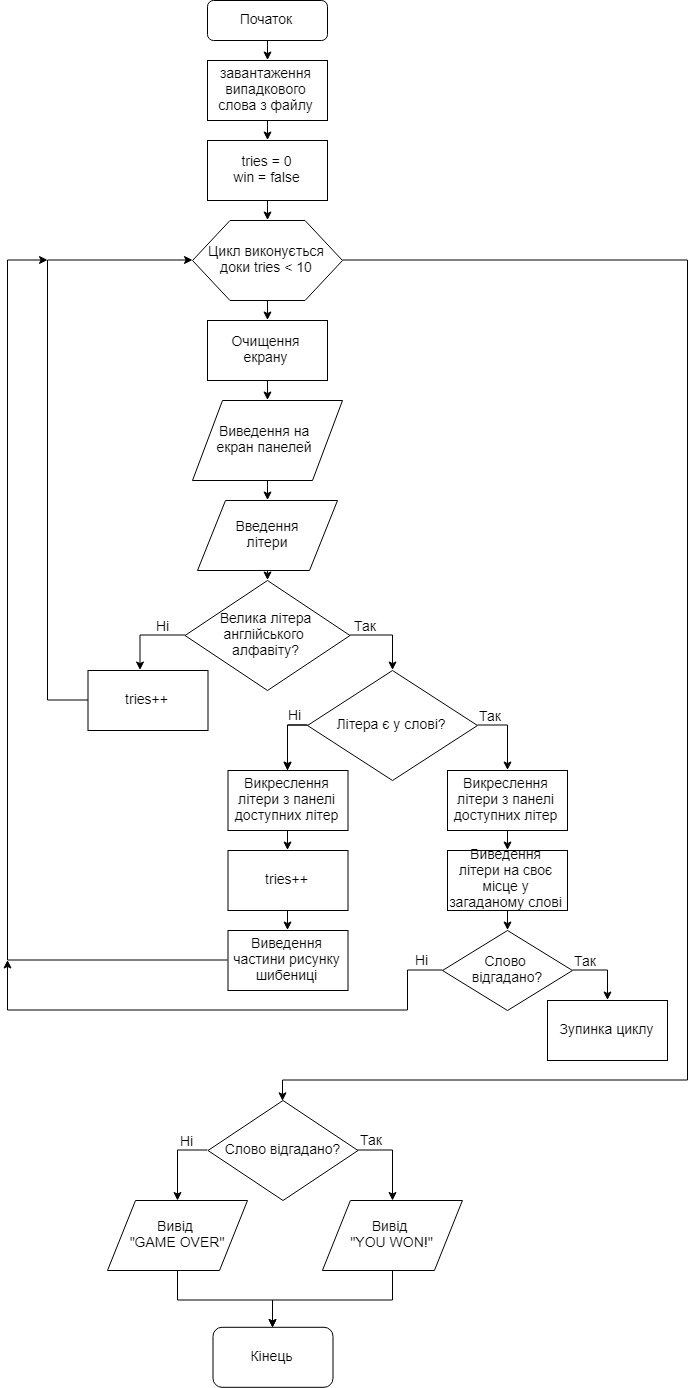


Рис. А.1. Блок- схема програми

**Додаток Б.1. Код програми**

#include <iostream>

#include <vector>

#include <fstream>

#include <string>

#include <time.h>

using namespace std;

void PrintMessage(string message, bool printtop = true, bool printbottom = true)

{

if (printtop) //створення верхньої частини панелі

{

cout << "+----------------------------------------------------------------------------------+" << endl;

cout << "|";

}

else

{

cout << "|";

}

bool front = true; //допоміжна змінна

for (int i = message.length(); i < 82; i++) //розміщення тексту по //центру панелі, 82 - оптимальна довжина панелі для комфортного читання

{

if (front)

{

message = " " + message;

}

else

{

**Додаток Б**

message = message + " ";

}

front = !front;

}

cout << message.c\_str();

if (printbottom) //створення нижньої частини панелі

{

cout << "|" << endl;

cout << "+----------------------------------------------------------------------------------+" << endl;

}

else

{

cout << "|" << endl;

}

}

void Rules(string rule) // вивід правил

{

bool front = true; //допоміжна змінна

for (int i = rule.length(); i < 82; i++)

{

if (front)

{

rule = " " + rule;

}

else

{

rule = rule + " ";

}

**Додаток Б**

front = !front;

}

cout << "|";

cout << rule.c\_str();

cout << "|" << endl;

}

void Draw(int guess=0)

{

if (guess >= 1) //одна помилка і більше

PrintMessage("|", false, false);

else

PrintMessage("", false, false);

if (guess >= 2) //дві помилки і більше

PrintMessage("|", false, false);

else

PrintMessage("", false, false);

if (guess >= 3) //три помилки і більше

PrintMessage("O", false, false);

else

PrintMessage("", false, false);

if (guess == 4) //чотири помилки

PrintMessage("/ ", false, false);

if (guess == 5) //п'ять помилок

PrintMessage("/|", false, false);

if (guess >= 6) //шість помилок і більше

PrintMessage("/|\\", false, false);

else

PrintMessage("", false, false);

if (guess >= 7) //сім помилок і більше

**Додаток Б**

PrintMessage("|", false, false);

else

PrintMessage("", false, false);

if (guess == 8) //вісім помилок

PrintMessage("/ ", false, false);

if (guess >= 9) //дев'ять помилок і більше

PrintMessage("/ \\", false, false);

else

PrintMessage("", false, false);

}

void PrintLetters(string input, char from, char to)

{

string s; //майбутній рядок з літерами

for (char i = from; i <= to; i++)

{

if (input.find(i) == string::npos) //якщо літера не знайдена у //введеному користувачем рядку

{

s += i; //вивід незнайденої літери на екран

s += " ";

}

else //якщо літера знайдена

s += " "; //вивід порожнього місця

}

PrintMessage(s, false, false); //вивід

}

void PrintAvailableLetters(string taken) //string taken - рядок вводу

{

**Додаток Б**

PrintMessage("Available letters", true, false); //вивід назви панелі з //невикористаними літерами

PrintLetters(taken, 'A', 'M'); //вивід невикористаних літер у проміжку //від А до М

PrintLetters(taken, 'N', 'Z'); //вивід невикористаних літер у проміжку //від N до Z

}

bool PrintWordAndCheckWin(string word, string guessed)

{

bool won = true;

string s; //рядок із загаданим словом

for (int i = 0; i < word.length(); i++)

{

if (guessed.find(word[i]) == string::npos) //якщо введена літера //не міститься у загаданому слові

{

won = false;

s += "\_ "; //вивід порожнього місця

}

else //якщо введена літера міститься у загаданому слові

{

s += word[i]; //вивід відгаданої літери

s += " ";

}

}

PrintMessage(s, false); //вивід панелі із загаданим словом

return won;

}

string LoadRandomWord(string path) //загадування слова

**Додаток Б**

{

string word; //загадане слово

vector<string> v; //динамічний масив типу string

ifstream reader(path); //путь до файлу

if (reader.is\_open()) //якщо файл відкритий

{

while (std::getline(reader, word))

v.push\_back(word); //додаємо слова з файлу до //динамічного масиву

int randomline = rand() % v.size(); //вибір випадкового слова з //динамічного масиву

word = v.at(randomline); //word набуває значення ссилки на //елемент randomline динамічного масиву

reader.close(); // закриття файлу

}

return word;

}

int TriesLeft(string word, string guessed) //підрахунок кількості помилок

{

int error = 0;

for(int i = 0 ; i < guessed.length(); i++)

{

if(word.find(guessed[i]) == string::npos)

error++;

}

return error;

}

**Додаток Б**

int main()

{

srand(time(0));

string guesses; //рядок використаних літер

string wordToGuess;

wordToGuess = LoadRandomWord("words.txt"); //загадане слово

int tries = 0; //кількість помилок

bool win = false;

do

{

system("cls"); //очищення екрану

PrintMessage("HANG MAN"); //вивід панелі з назвою гри

Rules("Purpose of the game: guess the word spoken by the computer."); //вивід першого рядка правил

Rules("Rules of the game: You can choose any letter."); //вивід //другого рядка правил

Rules("If there is such a letter, all similar letters in the word will be opened."); //вивід третього рядка правил

Rules("YOU CAN MAKE ONLY 10 MISTAKES!"); //вивід //четвертого правила

PrintMessage("YOU MUST USE ONLY BIG ENGLISH LETTERS!",false); //вивід умови користування та закриття панелі з правилами

Draw(tries); //малювання шибениці

PrintAvailableLetters(guesses); //вивід невикористаних літер

PrintMessage("Guess the word"); //вивід назви панелі із //загаданим словом

win = PrintWordAndCheckWin(wordToGuess, guesses); //змінна //win приймає значення функції PrintWordAndCheckWin

if (win)

**Додаток Б**

break; //завершення роботи циклу

char x;

cout << ">";

cin >> x; //ВВЕДЕННЯ ЛІТЕРИ

if (guesses.find(x) == string::npos) //якщо рядок не містить //введеної літери

guesses += x; //додання використаної літери до рядка

tries = TriesLeft(wordToGuess, guesses); //підрахунок помилок

} while (tries < 10); //цикл виконується, поки кількість помилок не //більше 10

if (win) //якщо слово відгадане

PrintMessage("YOU WON!");

else //якщо слово не відгадане

PrintMessage("GAME OVER");

system("pause");

getchar();

return 0;

}

**Додаток В.1. Файл програми**



Рис. В.1. Файл words.txt