**Лабораторна робота №8**

**Тема. Проектування ігри «Шибениця» за допомогою блок-схем**

**Мета роботи**. Освоїти ASCII-графіку. Освоїти проектування гри за допомогою блок-схем. Списки, оператор in, методи, строкові методи split(), lower(), upper(), startswith() і endswith(), інструкції elif.

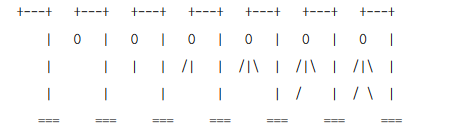
**Зміст**.

1. Вивчення відомостей про ASCII-графіку.
2. Освоїти проектування гри за допомогою блок-схем.
3. Отримання результату.

**Ключові положення.**

Шибениця - це гра для двох, в якій один гравець загадує слово (це буде комп’ютер) і малює на сторінці окремі порожні клітини для кожної літери. А другий гравець намагається вгадати букви, які можуть бути в даному слові, а потім і все слово. Якщо другий гравець правильно вгадує букву, перший гравець вписує її в відповідну порожню клітину. А якщо помиляється, перший гравець малює одну з частин тіла повішеного чоловічка. Щоб перемогти, другий гравець повинен вгадати всі букви в слові до того, як повішений чоловічок буде повністю намальований.

Графіка для гри «Шибениця» - це символи клавіатури, надруковані на екрані. Такий вид графіки називається ASCII-графіка. Нижче показано шибеницю, яка намальована за допомогою ASCII-графіки:



Блок-схема являє собою діаграму, яка відображатиме серію кроків у вигляді полів (блоків), пов'язаних один з одним стрілками. Кожне поле - це крок, а стрілки показують можливі кроки. На рис.1 показана повна блок-схема гри «Шибениця».

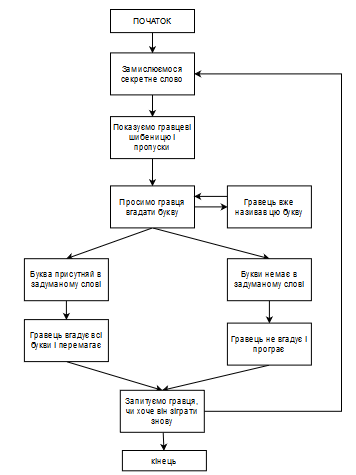


Рис.1

Підключимось до бібліотеки випадкових чисел за допомогою команди

import random

Рядки з 2 по 37 - це одна довга інструкція присвоювання для змінної HANGMAN\_PICS.

import random

HANGMAN\_PICS= ['''

+---+

|

|

|

===''','''

+---+

0 |

|

|

===''','''

+---+

0 |

| |

|

===''','''

+---+

0 |

/| |

|

===''','''

+---+

0 |

/|\ |

|

===''','''

+---+

0 |

/|\ |

/ |

===''','''

+---+

0 |

/|\ |

/ \ |

===''']

words = 'аист акула бабуин барсук бобр бык верблюд'.split()

У цьому рядку коду гри використовується метод split (). Код виглядає довгим, але, насправді, це просто рядок слів, розділених пробілами, з викликом методу split () в кінці. Метод split () створює список, в якому кожне слово з рядка стає окремим елементом списку words.

def getRandomWord(wordList):

# Ця функція повертає випадковий рядок з переданого списку

wordIndex = random.randint(0, len(wordList) - 1)

return wordList[wordIndex]

У рядку визначається функція getRandomWord (). Значення елементів списку передаються аргументу wordlist в якості параметрів. Ця функція повертає одиничне секретне слово з списку words.

wordIndex = random.randint (0, len (wordList) - 1) З списку в якості значення змінної wordIndex зберігається елемент з випадковим індексом. Випадковий вибір елемента проводиться функцією randint () з двома аргументами. Перший аргумент – 0 а другий - значення виразу len (wordList) - 1, що визначає останній можливий індекс. Змінна wordIndex зберігає випадковий індекс зі списку, переданого за допомогою параметра wordList. return wordList[wordIndex] - повертає зі списку wordList значення елемента з відповідним індексом, який зберігається як ціле число в wordIndex.

**Відображення ігрового поля для гравця**

Далі необхідна функція промальовування ігрового поля гри «Шибениця». На ньому має відображатися кількість введених гравцем букв, вгаданих як вірно, так і помилково.

**missedLetters** - Рядок букв, які гравець назвав, але їх немає в загаданому слові; **correctLetters** - Рядок букв, які гравець вгадав в загаданому слові; **secretWord -** Рядок з загаданим словом, яке гравець намагається відгадати.

def displayBoard(missedLetters, correctLetters, secretWord):

print(HANGMAN\_PICS[len(missedLetters)])

print()

print('Ошибочные буквы:', end=' ')

for letter in missedLetters:

print(letter, end= ' ')

print()

blanks = '\_' \* len(secretWord)

for i in range(len(secretWord)): # замінює пропуски відгаданими буквами

if secretWord[i] in correctLetters:

blanks = blanks[:i] + secretWord[i] + blanks[i+1:]

for letter in blanks: # Показує секретне слово з пробілами між буквами print(letter, end=' ')

print()

Спочатку функція print () викликає відображення ігрового поля. Глобальна змінна HANGMAN\_PICS містить список строкових змінних для промальовування всіх можливих етапів гри.

Код HANGMAN\_PICS [0] відображає порожню «шибеницю», код HANGMAN\_PICS [1] показує голову.

Число букв, яке зберігається в змінної missedLetters, відображає кількістю неправильних припущень, зроблених гравцем. Для визначення цього числа викликається функція len (missedLetters). Тобто якщо значення змінної missedLetters одно, наприклад, 'лелека', то код len ('лелека') поверне 6. Код HANGMAN\_PICS [6] відобразить повішеного, відповідного 6 промахів.

У циклі for відбувається перебір всіх сімволів змінної missedLetters і виведення їх на екран. Вираз end = '' заміщає символ нового рядка, який є в кінці кожного рядка, - одиничним пропуском. Наприклад, якщо значення missedLetters одно 'аизх', то такий цикл for виведе на екран а и з х.

Інша частина коду функції displayBoard () виводить на екран літери і формує рядок - секретне слово, в якому ще не угадані букви заміщені пробілами. Це досягається за допомогою функції range () і зрізу списку.

**Вивід секретного слова з пробілами**.

Строкова змінна blanks, з підкресленнями по числу букв секретного слова. Спочатку потрібно створити рядок з символів нижнього підкреслення для всіх букв секретного слова. Потім замінити прогалини цими символами. Наприклад, для секретного слова 'ворон' порожній рядок-підкреслення виглядає так: '\_ \_ \_ \_ \_' (п'ять підкреслень).

У циклі for виробляє послідовний перебір всіх букв секретного слова і заміщає підкреслення буквами, що містяться в змінної correctLetters.

def getGuess(alreadyGuessed):

# Повертає букву, введену гравцем.

# Ця функція перевіряє, що гравець ввів тільки одну букву і нічого більше

while True:

print('Введить букву.')

guess = input()

guess = guess.lower()

if len(guess) !=1:

print('Введить букву.')

elif guess in alreadyGuessed:

print('Ви вже називали цю букву. Назвить іншу.')

elif guess not in 'абвгдеежзийклмнопрстуфхцчшщъьэюяы':

print(Будь ласка, введіть ЛІТЕРУ.')

else:

return guess

Отримання припущень гравця при виконанні функції getGuess () гравець може ввести передбачувану букву. Ця функція повертає букву, запропоновану гравцем, у вигляді рядка. Далі функція getGuess () перевіряє допустимість введеного символу, перш ніж передати його в функцію.

Рядок букв, запропонованих гравцем, передається в якості аргументу в параметр alreadyGuessed функції. Потім функція getGuess () просить гравця ввести одну букву. Цю букву функція getGuess () поверне як своє значення.

**Перевірка допустимості припущення гравця**

Змінна guess містить букви, запропоновані гравцем. Програма повинна упевнитися, що вводяться символи допустимі, це повинна бути тільки одна буква, при цьому не повинні застосовуватися раніше. Якщо ця умова не виконується, цикл повертається до початку і знову запитує букву.

if len(guess) !=1: перевіряється, що введено не більше одного символу,

elif guess in alreadyGuessed: перевіряє, що запропонована буква не міститься в змінної alreadyGuessed,

elif guess not in 'абвгдеежзийклмнопрстуфхцчшщъьэюяы': перевіряє, що введений символ стандартного російського алфавіту. Якщо хоча б одна умова не виконується, гравцеві пропонується ввести іншу букву. Якщо виконані всі умови, програма переходить до виконання коду блоку і повертає значення запропонованої букви в рядку return guess.

Пропозиція гравцеві зіграти зановоФункція playAgain () містить лише виклик функції print () і инструк-цію return.

def playAgain():

# Ця функція повертає значення True, якщо гравець хоче зіграти заново;

#в іншому випадку повертає False

print('Хочете зіграти ще? (так чи ні)')

return input().lower().startswith('т')

Функція playAgain () дозволяє гравцеві відповісти «Так» або «Ні» на пред-ложение ще одного раунду. Гравець повинен ввести Так, так, Т або що-небудь ще, що починається з букви т, і це значення буде означати «Так». Якщо гравець ввів ТАК, функція input () повертає значення 'ТАК'. Вираз 'ТАК'.lower () повертає рядок в нижньому регістрі (маленькими літерами). Тобто значення 'ТАК' перетворюється в 'так'. Но, крім цього, проводиться ще й виклик методу startswith (' т '). Ця функція повертає True, якщо викликає рядок починається з указаного параметра, або False, якщо це не так. Наприклад, вираз 'так'.startwith (' т ') повертає значення True.

**Ігровий цикл**

Основна частина коду програми «Шибениця» відображає на екрані назву гри, містить кілька змінних і запускає цикл while. У цьому розділі розглянемо послідовне виконання решти програмного коду

print(‘Ш И Б Е Н И Ц Я ')

missedLetters = ''

correctLetters = ''

secretWord = getRandomWord(words)

gameIsDone = False

Код print(‘Ш И Б Е Н И Ц Я ') - викликає функцію print (), яка виводить на екран заголовок гри в момент її запуску. Потім строкової змінної missedLetters присвоюється порожнє значення, змінної correctLetters теж присвоюється порожнє значення, так як гравець не запропонував ще ніяких букв.В рядку secretWord = getRandomWord(words)- викликається функція getRandomWord (words), яка вибирає випадковим чином секретне слово з спіска. Код gameIsDone = False - привласнює змінної gameIsDone значення False. Код присвоїть цієї змінної значення True, коли надійде сигнал завершення гри, і запропонує гравцеві зіграти заново.

Виклик функції displayBoard () Частина програми складається з циклу while. Умова циклу завжди істинно, а це значить, що він буде виконуватися нескінченно довго, до тих пір, поки не буде ініційовано виконання інструкції break.

while True:

displayBoard(missedLetters, correctLetters, secretWord)

# Дозволяє гравцеві ввести букву

guess = getGuess(missedLetters + correctLetters)

if guess in secretWord:

correctLetters = correctLetters + guess

foundAllLetters = True

for i in range(len(secretWord)):

if secretWord[i] not in correctLetters:

foundAllLetters = False

break

if foundAllLetters:

print('Так! Секретне слово - "' + secretWord + '"! Ви вгадали!')

gameIsDone = True

else:

missedLetters = missedLetters + guess

# Перевіряє, чи перевищив гравець ліміт спроб і програв

if len(missedLetters) == len(HANGMAN\_PICS)-1:

displayBoard(missedLetters, correctLetters, secretWord)

print('Ви вичерпали усі спроби!\nНе вгадано букв: ' + str(len(missedLetters)) + ' та вгадано букв: ' + str(len(correctLetters)) + '. Было загадано слово "' + secretWord + '".')

gameIsDone = True

if gameIsDone:

if playAgain():

missedLetters= ''

correctLetters = ''

gameIsDone = False

secretWord = getRandomWord(words)

else:

break

Код викликає функцію displayBoard (), передаючи їй значення трьох змінних missedLetters, correctLetters, secretWord. Залежно від того, скільки букв гравець вгадав правильно і скільки разів помилився, ця функція виводить на екран відповідне зображення «повішеного».

**Введення гравцем букви, що вгадується**

Далее guess = getGuess(missedLetters + correctLetters) - викликається функція getGuess (), щоб гравець міг ввести букву, що вгадується.

У функцію передається параметр alreadyGuessed для визначення – чи вводив гравець таку букву раніше або ще ні.

guess = getGuess(missedLetters + correctLetters)- конкатенуються рядкові змінні missedLetters і correctLetters, а потім передає результат як аргумент параметру alreadyGuessed.

**Перевірка наявності букви в секретному слові**

Якщо запропонована буква є в секретному слові, вона додається в кінець строкової змінної correctLetters. Та додає нове значення в змінну correctLetters.

if guess in secretWord:

correctLetters = correctLetters + guess

Цикл починається в припущенні, що всі букви секретного слова вгадані. Але в рядку foundAllLetters = False, у процесі виконання циклу, значення змінної foundAllLetters змінюється на False, як тільки виявлена перша буква з змінної secretWord, що не міститься в змінної correctLettrs. Якщо всі букви секретного слова виявлені, гравцеві повідомляється про його перемогу, а змінна gameIsDone приймає значення True.

**Обробка помилкових припущень**

Неправильно вгадані букви додаються в строкову змінну missedLetters в рядку missedLetters = missedLetters + guess. Це відбувається так само як в рядку correctLetters = correctLetters + guess з буквами, вгаданими вірно.

Перевірка - чи не програв гравець. Кожен раз, коли гравець вводить неправильну букву, вона додається в змінну missedLetters. Таким чином, довжина значення змінної missedLetters (або в коді - len (missedLetters)) стає дорівнює числу помилкових припущень.

Змінна HANGMAN\_PICS містить сім рядків з ASCII-символами рисунка. Таким чином, якщо довжина рядка missedLetters дорівнює len (HANGMAN\_PICS) - 1 (тобто 6), гравець вичерпав ліміт припущень. Якщо малюнок «повішеного» завершено - гравець програв.

**Завершення або перезавантаження ігри**

В не залежності від перемоги або програшу гра повинна запитати гравця - чи хоче він зіграти знову. Функція playAgain () обробляє отриману відповідь ( «Так» або «Ні») в рядку if playAgain(). Якщо гравець хоче почати гру заново, значення змінних missedLettersі correctLetters треба обнулити, змінної gameIsDone привласнити значення False і вибрати нове секретне слово. Після того як виконання циклу while повертається до початку, ігровий інтерфейс, який відображається на екрані, перезавантажується і готовий до нової гри. Якщо гравець не ввів нічого, що починається з букви т, при запиті - чи хоче він зіграти заново, умова стає хибною і виконується блок else.

Лабораторне завдання

* + - 1. 1. Змінити гру так, щоб один гравець вводив слово з клавіатури. Воно не відображалось на екрані, другий відгадував це слово.
      2. 2. Змінити шибеницю на відображення цифр від 9 до 0 по кількості поразок.

Програма

