

**Московский государственный технический
университет им. Н.Э. Баумана**

Факультет «Радиотехнический»
Кафедра «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по домашнему заданию
«Разработка игры "Виселица" на языке F#»

Выполнил:
студент группы РТ5-31Б
Романов Ю.И.
Подпись и дата:

Проверил:
к.т.н., доц. каф. ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Москва, 2025 г.

Описание задания

1) Выберите язык программирования (который Вы ранее не изучали) и (1) напишите по нему реферат с примерами кода или (2) реализуйте на нем небольшой проект (с детальным текстовым описанием).

2) Реферат (проект) может быть посвящен отдельному аспекту (аспектам) языка или содержать решение какой-либо задачи на этом языке.

3) Необходимо установить на свой компьютер компилятор (интерпретатор, транспилятор) этого языка и произвольную среду разработки.

4) В случае написания реферата необходимо разработать и откомпилировать примеры кода (или модифицировать стандартные примеры).

5) В случае создания проекта необходимо детально комментировать код.

6) При написании реферата (создании проекта) необходимо изучить и корректно использовать особенности парадигмы языка и основных конструкций данного языка.

7) Приветствуется написание черновика статьи по результатам выполнения ДЗ. Черновик статьи может быть подготовлен группой студентов, которые исследовали один и тот же аспект в нескольких языках или решили одинаковую задачу на нескольких языках.

Я выбрал язык программирования F# и разработал на нем консольную программу для игры «Виселица».

Текст программы

hangman.fs:

`open System`

```
// Список слов для угадывания
let words = [
    "функциональный"
    "программирование"
    "виселица"
    "компьютер"]
```

```

    "алгоритм"
    "структура"
    "переменная"
    "разработка"
    "интерфейс"
    "клавиатура"
]

// Состояние игры
type GameState = {
    Word: string
    GuessedLetters: Set<char>
    WrongAttempts: int
    MaxAttempts: int
}

// Инициализация новой игры
let newGame () =
    let random = Random()
    let word = words.[random.Next(words.Length)]
    {
        Word = word
        GuessedLetters = Set.empty
        WrongAttempts = 0
        MaxAttempts = 6
    }

// Отображение текущего состояния слова
let displayWord state =
    state.Word
    |> Seq.map (fun c =>
        if state.GuessedLetters.Contains(c) then c
        else '_')
    |> Seq.toArray
    |> String

// Проверка, угадано ли слово полностью
let isWordGuessed state =
    state.Word
    |> Seq.forall (fun c -> state.GuessedLetters.Contains(c))

// Отображение виселицы
let displayHangman attempts maxAttempts =
    let stages = [

```



```
"""
-----
| |
O |
/ \ |
/   |
-----
"""
"""
"""
-----
| |
O |
/ \ |
/ \ |
|
-----
"""
]
```

```
let index = min attempts (stages.Length - 1)
printfn "%os" stages.[index]
printfn "Ошибка: %d/%d" attempts maxAttempts

// Обработка ввода буквы
let processGuess state guess =
    let guessChar = Char.ToLower(guess)

    // Проверяем, была ли уже такая буква
    if state.GuessedLetters.Contains(guessChar) then
        printfn "Вы уже вводили букву '%c'" guessChar
        state
    else
        let newGuessed = state.GuessedLetters.Add(guessChar)

    // Проверяем, есть ли такая буква в слове
    if state.Word.Contains(guessChar) then
        printfn "Правильно! Буква '%c' есть в слове." guessChar
        { state with GuessedLetters = newGuessed }
    else
        printfn "Неправильно! Буквы '%c' нет в слове." guessChar
        {
            state with
                GuessedLetters = newGuessed
                WrongAttempts = state.WrongAttempts + 1
        }
```

```
    }

// Отображение использованных букв
let displayUsedLetters state =
    let usedLetters =
        state.GuessedLetters
        |> Set.toList
        |> List.sort

    if not usedLetters.IsEmpty then
        printf "Использованные буквы: "
        usedLetters |> List.iter (printf "%c ")
        printfn ""

// Основной игровой цикл
let rec gameLoop state =
    Console.Clear()

    printfn "==== Игра 'Виселица' ==="
    printfn ""

    displayHangman state.WrongAttempts state.MaxAttempts
    printfn ""

    let currentDisplay = displayWord state
    printfn "Слово: %s" currentDisplay
    printfn ""

    displayUsedLetters state
    printfn ""

    // Проверка условий окончания игры
    if isWordGuessed state then
        printfn "Поздравляем! Вы угадали слово: %s" state.Word
        printfn "Игра окончена!"
    elif state.WrongAttempts >= state.MaxAttempts then
        printfn "Игра окончена! Вы проиграли."
        printfn "Загаданное слово было: %s" state.Word
        displayHangman state.MaxAttempts state.MaxAttempts
    else
        printf "Введите букву: "
        let input = Console.ReadLine()

        if String.IsNullOrEmpty(input) then
            printfn "Пожалуйста, введите букву."
```

```

Console.ReadLine() > ignore
gameLoop state
else
    let guess = input.[0]
    let newState = processGuess state guess
    Console.ReadLine() > ignore
    gameLoop newState

// Запуск игры
let startGame () =
    printfn "Добро пожаловать в игру 'Виселица'!"
    printfn "Угадайте слово по буквам. У вас есть 6 попыток."
    printfn "Нажмите любую клавишу для начала игры..."
    Console.ReadKey() > ignore

    let initialState = newGame()
    gameLoop initialState

// Точка входа
[<EntryPoint>]
let main argv =
    startGame()
    0

```

Экранные формы с примерами выполнения программы

○ yuriy@192 code % dotnet run
Добро пожаловать в игру 'Виселица'!
Угадайте слово по буквам. У вас есть 6 попыток.
Нажмите любую клавишу для начала игры...
█

==== Игра 'Виселица' ===



Ошибка: 0/6

Слово: _____

Введите букву: █

==== Игра 'Виселица' ====



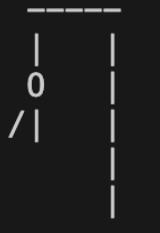
Ошибок: 3/6

Слово: клавиа__а

Использованные буквы: а в и к л п ф я

Введите букву:

==== Игра 'Виселица' ====



Ошибок: 3/6

Слово: клавиатура

Использованные буквы: а в и к л п р т у ф я

Поздравляем! Вы угадали слово: клавиатура
Игра окончена!