Greek Wordle

1.Εισαγωγή & Motivation

Το Wordle είναι ένα παιχνίδι στο οποίο πρέπει να μαντέψεις μια λέξη πέντε γραμμάτων. Η λέξη είναι η ίδια παγκοσμίως και αλλάζει καθημερινά. Το παιχνίδι αναπτύχθηκε από τον Josh Wardle το 2021 και αφού έγινε viral, το 2022 αποκτήθηκε από την New York Times. Δεν απαιτεί εφαρμογή και μπορεί να παιχτεί στον browser.

Συχνά το παίζουμε στα διαλείμματα των μαθημάτων. Οπότε μας κίνησε το ενδιαφέρον η ιδέα να το υλοποιήσουμε μόνοι μας μέσω της γλώσσας C++ και μάλιστα στα ελληνικά.

2. Objective & Scope

- **Ποιοι είναι οι βασικοί στόχοι;**
    - Υλοποίηση ενός πλήρως λειτουργικού Wordle με ελληνικές λέξεις.
    - Παροχή ανατροφοδότησης (feedback) στον παίκτη για κάθε προσπάθεια.
    - Υποστήριξη πολλαπλών προσπαθειών και τερματισμός όταν βρεθεί η σωστή λέξη ή εξαντληθούν οι προσπάθειες.
- **Ποιες δυνατότητες περιλαμβάνονται;**
    - Εύκολη επεκτασιμότητα για προσθήκη περισσότερων λέξεων.
    - Αναγνώριση σωστών/λανθασμένων γραμμάτων και θέσεων.
    - Λειτουργία στο console χωρίς πρόσθετες εξαρτήσεις.

3. System Architecture

Το πρόγραμμα υλοποιεί μια κλασική τριεπίπεδη λογική:

Παρουσίαση (Console) – Η διεπαφή του χρήστη στηρίζεται αποκλειστικά στη Windows Console. Η συνάρτηση `Menu` σχεδιάζει σε κάθε γύρο ένα «πληκτρολόγιο» QWERTY με ελληνικά κεφαλαία γράμματα· κάθε χαρακτήρας εμφανίζεται με χρώμα που υποδηλώνει την τρέχουσα κατάστασή του (πράσινο, κίτρινο, γκρι). Για τον χρωματισμό αξιοποιείται το `SetConsoleTextAttribute` της βιβλιοθήκης `windows.h`, ενώ η μεταβλητή-χειριστής `hConsole` παραμένει ανοιχτή σε όλη τη διάρκεια εκτέλεσης, αποφεύγοντας περιττά system calls.

Λογική Παιχνιδιού – Ο βασικός βρόχος ζει στη `main`. Ένας εξωτερικός `while` ελέγχει αν ο χρήστης επιθυμεί νέο παιχνίδι και ένας εσωτερικός καταμετρά τις έως έξι προσπάθειες. Σε κάθε επανάληψη εκτελούνται, με αυτή ακριβώς τη σειρά, οι ρουτίνες:

1. `Menu` – εκτύπωση πληκτρολογίου και banner προσπάθειας
2. `GetAndCheckInput` – ανάγνωση, καθαρισμός, έλεγχος εγκυρότητας λέξης
3. `CheckGuess` – υπολογισμός και εκτύπωση χρώματος για κάθε γράμμα
4. Έλεγχος νίκης ή ήττας· σε αρνητικό αποτέλεσμα αυξάνεται ο μετρητής `tries`.

Επίπεδο Δεδομένων / Βοηθητικών Συναρτήσεων –

- Το λεξικό φορτώνεται μία και μοναδική φορά στη `RandomWordSelection`: το αρχείο `5letterwords.txt` διαβάζεται με `wifstream` και κάθε μη κενή γραμμήαποθηκεύεται στον δυναμικό πίνακα `wordlist`. Στη συνέχεια μια τυχαία λέξη επιλέγεται μέσω `rand() % wordlist.size()`.
- Ο έλεγχος εγκυρότητας ελληνικών χαρακτήρων πραγματοποιείται στη `isWord`, η οποία επαληθεύει ότι όλα τα code points του input ανήκουν στα ranges U+0370–03FF ή U+1F00–1FFF.
- Η map `ColourOfLetters` (γράμμα → χρώμα) δρα ως «ενιαία πηγή αλήθειας» για την κατάσταση κάθε χαρακτήρα, διασφαλίζοντας ότι δεν προκύπτουν υποβιβασμοί (no downgrade)· ένα πράσινο γράμμα δεν γίνεται ποτέ κίτρινο ή γκρι σε μεταγενέστερη προσπάθεια.
- αποθηκεύεται στον δυναμικό πίνακα `wordlist`. Στη συνέχεια μια τυχαία λέξη επιλέγεται μέσω `rand() % wordlist.size()`.
- Ο έλεγχος εγκυρότητας ελληνικών χαρακτήρων πραγματοποιείται στη `isWord`, η οποία επαληθεύει ότι όλα τα code points του input ανήκουν στα ranges U+0370–03FF ή U+1F00–1FFF.
- Η map `ColourOfLetters` (γράμμα → χρώμα) δρα ως «ενιαία πηγή αλήθειας» για την κατάσταση κάθε χαρακτήρα, διασφαλίζοντας ότι δεν προκύπτουν υποβιβασμοί (no downgrade)· ένα πράσινο γράμμα δεν γίνεται ποτέ κίτρινο ή γκρι σε μεταγενέστερη προσπάθεια.

4 . Τεχνολογίες που χρησιμοποιήθηκαν

- **Γλώσσα C++17** – επιλέχθηκε για τη διαθεσιμότητα της Standard Library, τη διαχείριση «στενής» μνήμης και την ταχύτητα μεταγλώττισης/εκτέλεσης.
- **Βιβλιοθήκη Windows API (`windows.h`)** – δίνει πρόσβαση στην κλήση `SetConsoleTextAttribute`, που είναι ο απλούστερος τρόπος για 16-χρωμη έξοδο στη γραμμή εντολών των Windows.
- **Διαχείριση Unicode** – μέσω `_setmode(_O_U16TEXT)` επιβάλλεται χρήση UTF-16 wide streams· σε συνδυασμό με `setlocale(LC_ALL,"el_GR.UTF-8")` εξασφαλίζεται σωστή απεικόνιση και εισαγωγή ελληνικών χαρακτήρων.
- **STL Containers** – `vector<wstring>` για το λεξικό (O(1) πρόσβαση με δείκτη) και `map<wchar_t,int>` ώστε ο έλεγχος / ενημέρωση χρώματος κάθε γράμματος να γίνεται σε O(log n) (n = 24).
- **Αφαίρεση τόνων** – υλοποιείται χειροκίνητα με ένα `switch` που χαρτογραφεί τους 7 πιθανούς τονισμένους χαρακτήρες στα αντίστοιχα άτονα κεφαλαία· έτσι αποφεύγεται η ανάγκη για εξωτερική βιβλιοθήκη ICU.

5 . Κώδικας & Υλοποίηση

Η λογική αξιολόγησης βρίσκεται στη συνάρτηση `CheckGuess`. Η διαδικασία συντελείται σε δύο βήματα:

1. **Πλήρης ταύτιση (exact match)** – γίνεται μια πρώτη διέλευση των πέντε χαρακτήρων· εάν `guess[i] == CorrectWord[i]`, το αντίστοιχο στοιχείο του πίνακα `colour` λαμβάνει την τιμή 10 (πράσινο) και σημειώνεται στο `matched` ότι η θέση αυτή «καταναλώθηκε». Έτσι αποτρέπεται η διπλή καταμέτρηση ίδιων γραμμάτων.
2. **Μερική ταύτιση (present-but-misplaced)** – σε δεύτερη διέλευση ελέγχεται κάθε μη πράσινος χαρακτήρας μήπως απαντάται αλλού μέσα στη λέξη. Αν βρεθεί γράμμα που δεν έχει ήδη «μαρκαριστεί» ως matched, ο χρωματισμός γίνεται κίτρινος (τιμη 14) και η αντίστοιχη θέση του πίνακα `matched` κλειδώνει, ώστε να μη χρησιμοποιηθεί σε μελλοντική αντιστοίχιση. Όσοι χαρακτήρες απομένουν αχρωμάτιστοι χαρακτηρίζονται αυτόματα γκρι (τιμη 15).
3. Μετά τον υπολογισμό, η συνάρτηση:

   Εκτυπώνει στη γραμμή παιχνιδιού είτε το ίδιο το γράμμα (όταν είναι πράσινο/κίτρινο)

Ενημερώνει τη `ColourOfLetters`, λαμβάνοντας υπόψη τον κανόνα **no downgrade**: εάν ένα γράμμα έχει ήδη πάρει υψηλότερη «βαθμίδα» (πράσινο > κίτρινο > γκρι), το νέο χρώμα αγνοείται.

Παράλληλα, η συνάρτηση `GetAndCheckInput` επιλύει το πρόβλημα της ορθής εισαγωγής:

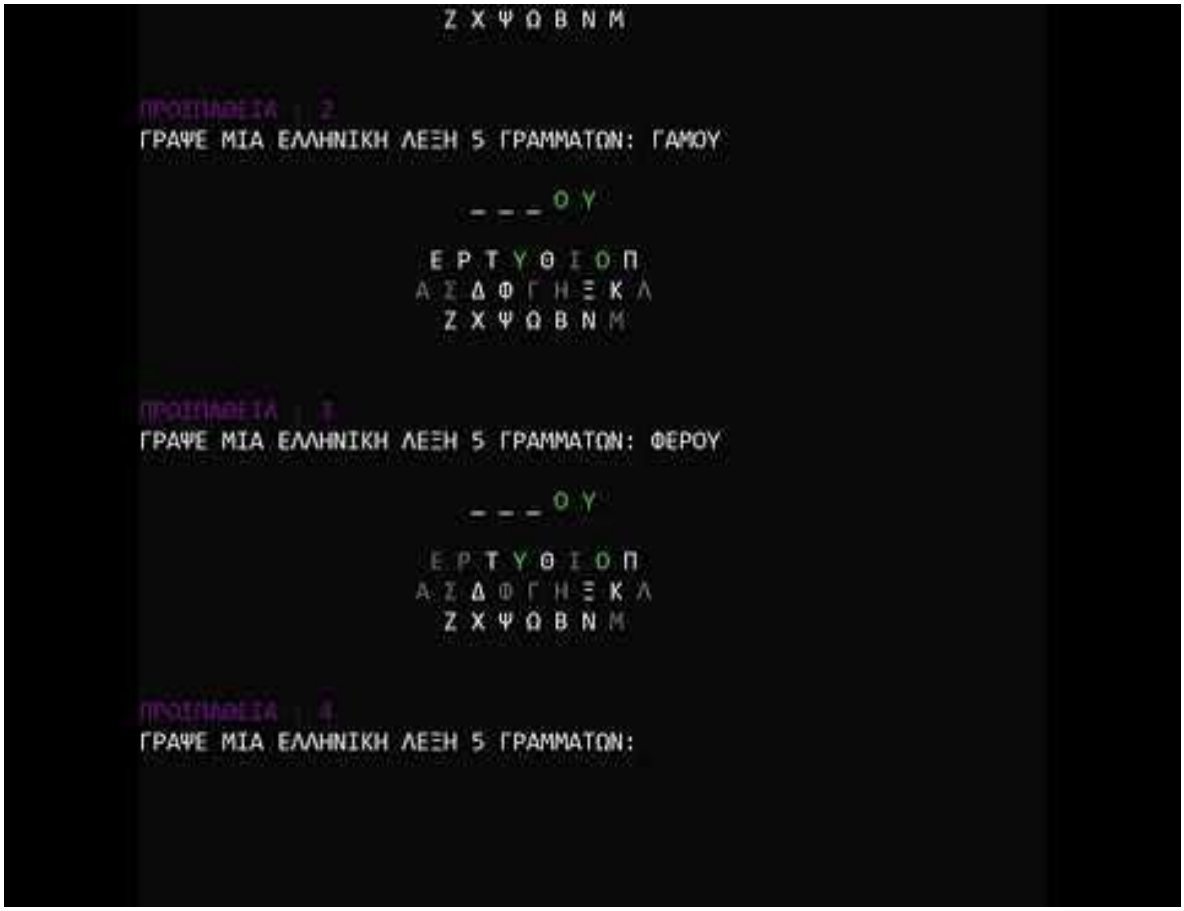
- Διαβάζει τη γραμμή ως wide string.
- Αφαιρεί κενά, μετατρέπει σε κεφαλαία, αποτονίζει και αντικαθιστά το τελικό «ς» με «Σ», ώστε ο έλεγχος να παραμένει case-insensitive και accent-insensitive.
- Εκτελεί Unicode range-check με το `isWord`.
- Τέλος, επιβεβαιώνει ότι η λέξη υπάρχει πράγματι στο `wordlist`. Αυτός ο έλεγχος προλαμβάνει «τυχαία» inputs που δεν ανήκουν στο πεδίο του παιχνιδιού.

Συνοψίζοντας, το πρόγραμμα επιτυγχάνει:

- **Ορθό Unicode I/O**, ακόμη και για χρήστες που πληκτρολογούν με τόνους.
- **Πιστή αναπαράσταση Wordle-scores** με ελάχιστο κώδικα.
- **Ελάχιστες εξαρτήσεις** και άμεση μεταγλώττιση στο πιο κοινό περιβάλλον ανάπτυξης C++ για Windows.

6 Αποτελέσματα demo

https://www.youtube.com/watch?v=kLCi0M5emQA



7 Comparison with AI generated code

Χαρακτηριστικα	AI Υλοποίηση	Δική μας Υλοποίηση
Εμφάνιση	Πολύ απλή, χωρίς χρώματα	Εντυπωσιακή, με χρώματα μέσω Windows API
Χειρισμός Εισόδου	Βασικός, χωρίς έλεγχο ύπαρξης λέξης σε λίστα	Προσεγμένος, ελέγχει εάν η λέξη υπάρχει στο αρχείο
Υποστήριξη Γλώσσας	Υποτυπώδης υποστήριξη ελληνικών	Πλήρης υποστήριξη UTF-16, τόνων, και ειδικών χαρακτήρων
Δομή Κώδικα	Μονολιθική, όλα σε `main()`	Δομημένη σε συναρτήσεις με διαχωρισμό ευθυνών

Επέκταση/Συντήρηση	Δύσκολη	Εύκολη, λόγω modular σχεδίασης
Οδηγίες & Διεπαφή	Δεν παρέχονται	Ενσωματωμένες οδηγίες, καθαρή διεπαφή χρήστη
Επαναληψιμότητα	Παίζεται μόνο μία φορά	Δυνατότητα επανάληψης παιχνιδιού

AI CODE

```cpp
#include <iostream>
#include <string>
#include <vector>
#include <ctime>
#include <cstdlib>
#include <algorithm>
using namespace std;

vector<wstring> greekWords = {L"ΜΗΛΟΙ", L"ΠΑΝΙΑ", L"ΣΑΒΒΑ", L"ΚΑΡΟΣ", L"ΝΕΡΟΙ"};

void ToUpper(wstring &word) {
    for (auto &ch : word) {
        ch = towupper(ch);
        if (ch == L'Ά') ch = L'Α';
        else if (ch == L'Έ') ch = L'Ε';
        else if (ch == L'Ή') ch = L'Η';
        else if (ch == L'Ί') ch = L'Ι';
        else if (ch == L'Ό') ch = L'Ο';
        else if (ch == L'Ύ') ch = L'Υ';
        else if (ch == L'Ώ') ch = L'Ω';
        else if (ch == L'ς') ch = L'Σ';
    }
}

void CheckGuess(const wstring &guess, const wstring &correct) {
    for (int i = 0; i < 5; ++i) {
        if (guess[i] == correct[i]) {
            wcout << L"[ " << guess[i] << L" ] "; // σωστό
        } else if (correct.find(guess[i]) != wstring::npos) {
            wcout << L"( " << guess[i] << L" ) "; // λάθος θέση
        } else {
            wcout << L"  " << guess[i] << L"   "; // δεν υπάρχει
        }
    }
    wcout << endl;
}

int main() {
    srand(time(0));
    wstring correctWord = greekWords[rand() % greekWords.size()];
    wstring guess;

    int attempts = 6;
    wcout << L"Βρες τη λέξη 5 γραμμάτων! Έχεις 6 προσπάθειες." << endl;

    while (attempts--) {
        wcout << L"\nΜάντεψε λέξη: ";
        getline(wcin, guess);
        ToUpper(guess);

        if (guess.length() != 5) {
            wcout << L"Δώσε λέξη 5 γραμμάτων!\n";
            attempts++;
            continue;
        }

        if (guess == correctWord) {
            wcout << L"ΜΑΝΑΔΑ! ΤΗΝ ΒΡΗΚΕΣ!\n";
            break;
        }

        CheckGuess(guess, correctWord);

        if (attempts == 0)
            wcout << L"\nΗ λέξη ήταν: " << correctWord << endl;
    }

    return 0;
```