



HoBeerSystem

ALGORYTMY I STRUKTURY DANYCH 2

Skład zespołu wraz z funkcjami:

Paweł Szczepankiewicz:

Koordynator, Programista, Tester, Autor dokumentacji

Kamil Nalewajski:

Programista, Tester, Strona graficzna

Konrad Zdziarski:

Programista, Tester, Strona graficzna

A large, horizontal, pink brushstroke graphic that tapers at both ends, resembling a paintbrush stroke. It is positioned on the right side of the slide, partially overlapping the text 'Zespół nr: 1'.

Zespół nr: 1

Cel i kontekst projektu

Projekt dla mieszkańców Shire — spokojnej krainy Hobbitów, znanej z piwa i porządku.

Rosnąca produkcja jęczmienia i browarów wymagała narzędzia do zarządzania i organizacji pracy.

Celem było wsparcie procesów logistycznych i informacyjnych w Shire.



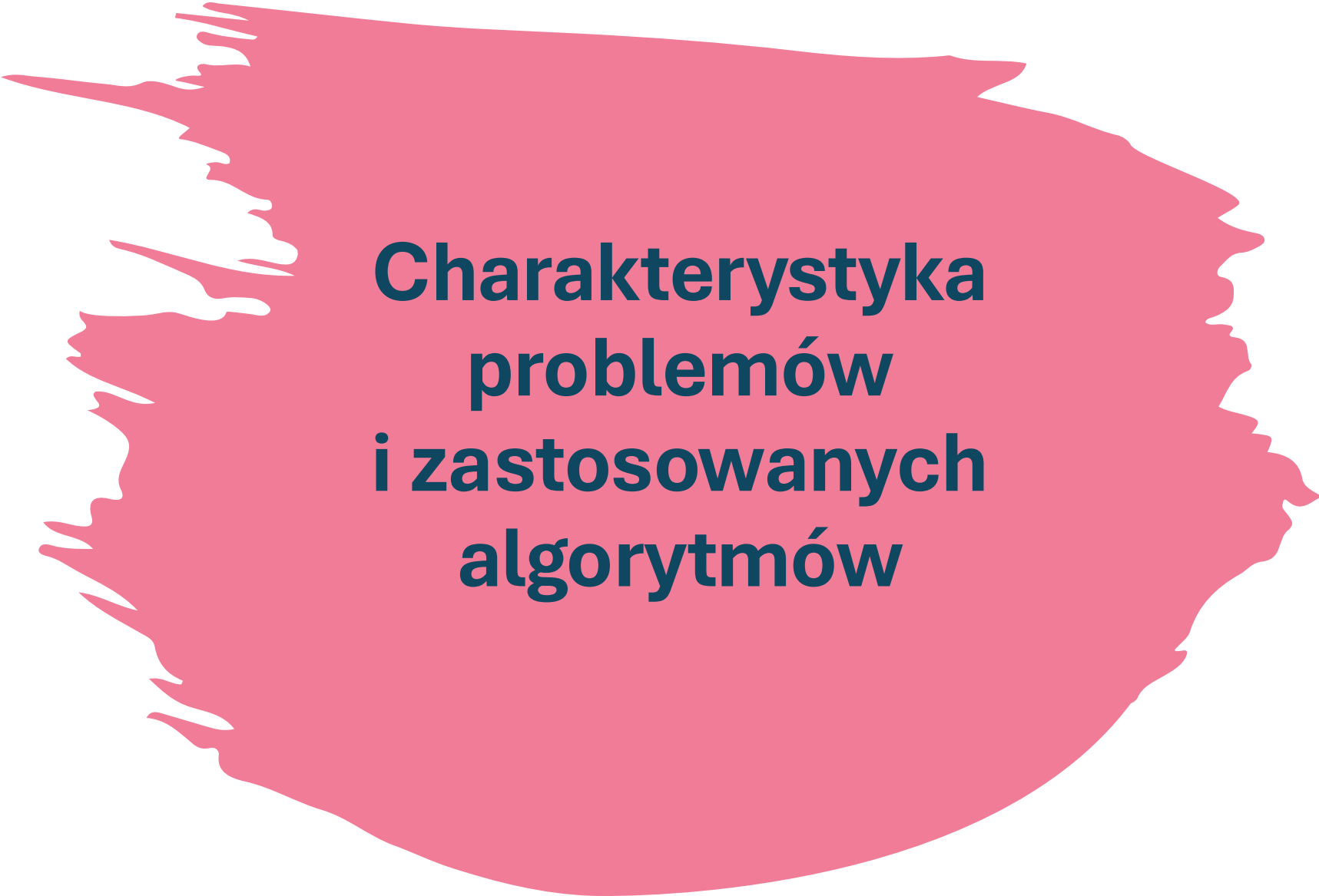
Kluczowe funkcje aplikacji

Graficzne
przedstawienie
pól, browarów
i karczm


Szybkie
wyszukiwanie
słów
kluczowych

Kompresję
i dekompresję
danych

Prosty
i intuicyjny
interfejs

A large, irregular pink brushstroke shape serves as a background for the text, centered on a white page. The brushstroke has a textured, hand-painted appearance with varying shades of pink and some darker edges.

Charakterystyka problemów i zastosowanych algorytmów



Problem 1

Reprezentacja danych o infrastrukturze Shire

Cel: Opisać infrastrukturę Shire (pola, browary, karczmy, drogi).

Jak: Użyto struktur i kontenerów STL do efektywnego przechowywania i zarządzania danymi.

Dlaczego: Zapewniały przejrzystość, elastyczność i wydajność w operowaniu dużą ilością danych.

Problem 2

Maksymalizacja przepływu piwa do karczm

Cel: Maksymalnie zwiększyć transport piwa do karczm przy ograniczeniach dróg.

Jak: Zastosowano algorytm Edmondsa-Karpa do znalezienia maksymalnego przepływu w grafie.

Dlaczego: Algorytm jest prosty, poprawny i efektywny dla średniej wielkości sieci.

Problem 3

Minimalizacja kosztu transportu przy maksymalnym przepływie

Cel: Zminimalizować koszty transportu przy zachowaniu maksymalnego przepływu.

Jak: Wykorzystano algorytm Successive Shortest Path i Dijkstrę do optymalizacji kosztów.

Dlaczego: Pozwala skutecznie rozwiązać problem minimalnego kosztu przepływu z dodatnimi wagami.

Problem 4

Przetwarzanie danych przestrzennych – analiza ćwiartek pól

Cel: Określić kształt i powierzchnię ćwiartek pól uprawnych.

Jak: Użyto algorytmu Grahama do wypukłej otoczki i wzoru shoelace do obliczenia pola.

Dlaczego: Algorytm Grahama jest dokładny i szybki do analizy geometrii płaskiej.

Problem 5

Wyszukiwanie słów w tekście

Cel: Szybko wyszukiwać słowa w tekstach (np. „piwo”, „browar”).

Jak: Porównano algorytmy: naiwny, KMP, Rabina-Karpa, Trie i Boyera-Moorea, by znaleźć najlepszy.

Dlaczego: Każdy algorytm ma różne zalety, pozwalając dobrać najlepszy w zależności od potrzeb.

Problem 6

Kompresja danych z użyciem algorytmu Huffmana

Cel: Zmniejszyć rozmiar danych tekstowych przy zachowaniu treści.

Jak: Zastosowano algorytm Huffmana do bezstratnej kompresji.

Dlaczego: Huffman jest prosty, optymalny i skuteczny przy ograniczonych zasobach.

Problem 7

Wizualizacja i interfejs użytkownika

Cel: Umożliwić intuicyjną obsługę i wizualizację wyników.

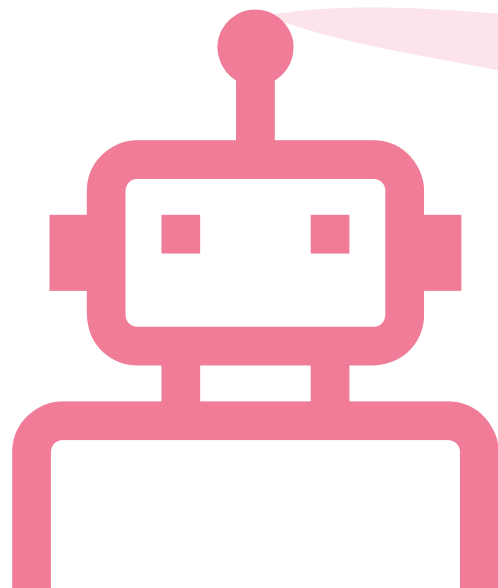
Jak: Stworzono graficzny interfejs w Qt, integrujący dane i algorytmy.

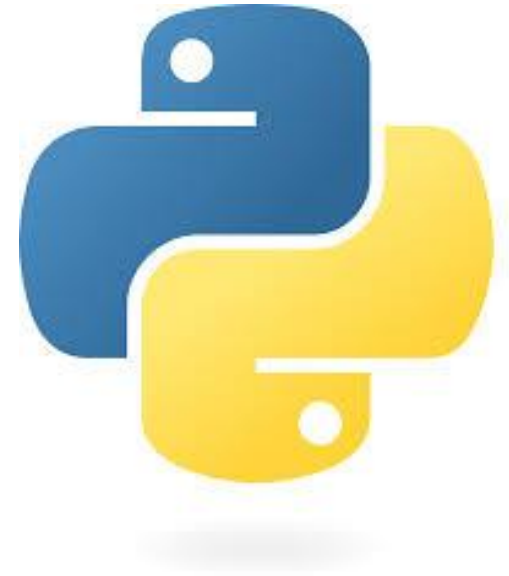
Dlaczego: Qt oferuje wygodne narzędzia do tworzenia estetycznych i funkcjonalnych GUI.

Proces tworzenia aplikacji

Projekt rozpoczęliśmy od ustalenia celów i podziału zadań. Pierwsze etapy, jak struktury danych i algorytmy przepływu, przebiegały sprawnie i zgodnie z harmonogramem.

Mimo trudności, zakończyliśmy projekt sukcesem, tworząc funkcjonalne narzędzie wspierające pracę hobbitów.





***Wykorzystane
Technologie***

Podsumowanie projektu

Projekt spełnił założone cele – usprawnia logistykę, wyszukiwanie tekstów, kompresję i wizualizację.

Praca zespołowa była kluczem do pokonania wyzwań.

Na przyszłość planujemy optymalizację wydajności i rozbudowę interfejsu o dodatkowe funkcje.



***Dziękujemy
za uwagę!***

