|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

**Wprowadzenie teoretyczne**

Funkcje w języku C++ są podstawowym elementem strukturalnym programu, pozwalającym na organizację kodu w logiczne i łatwe do zarządzania bloki. Dzięki funkcjom można wielokrotnie używać tych samych fragmentów kodu bez potrzeby jego powielania, co wpływa na przejrzystość i modularność programu. W programowaniu funkcje pełnią kluczową rolę, umożliwiając realizację zadań takich jak obliczenia, przetwarzanie danych, a także zarządzanie strukturami i algorytmami.

W ramach niniejszego opracowania przedstawiono zastosowanie funkcji do rozwiązywania różnorodnych problemów, takich jak generowanie liczb pseudolosowych, przeliczanie jednostek ciśnienia oraz obliczanie wartości różnych ciągów liczbowych. Każdy z tych problemów wymaga specyficznego podejścia i zastosowania odpowiednich algorytmów.

**Ciągi liczbowe**

1. **Ciąg arytmetyczny** – ciąg, w którym każdy kolejny wyraz jest sumą wyrazu poprzedzającego i stałej różnicy rr. Wzór ogólny:

an+1=an+ran+1​=an​+r

1. **Ciąg geometryczny** – ciąg, w którym każdy kolejny wyraz jest iloczynem wyrazu poprzedniego i ustalonego ilorazu qq. Wzór ogólny:

an+1=an⋅qan+1​=an​⋅q

1. **Ciąg Fibonacciego** – ciąg definiowany rekurencyjnie, w którym pierwszy wyraz wynosi 0, drugi 1, a każdy kolejny jest sumą dwóch poprzednich:

F0=0,F1=1,Fn=Fn−1+Fn−2dla n>1F0​=0,F1​=1,Fn​=Fn−1​+Fn−2​dla n>1

**Zadania**

* **Zadanie 1:** Symulacja losowania liczb „Totolotka” z wykorzystaniem funkcji pseudolosowych i opóźnienia wyświetlania liczb.
* **Zadanie 2:** Konwersja jednostek ciśnienia z Pascali na inne jednostki za pomocą funkcji konwertera.
* **Zadanie 3:** Sprawdzenie, czy trzy liczby tworzą trójkę pitagorejską.
* **Zadania 4-6:** Obliczanie elementów ciągów arytmetycznych, geometrycznych oraz Fibonacciego przy użyciu funkcji i tablic.

**Zadanie 1**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

**Zadanie 2**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

**Zadanie 3**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

**Zadanie 4**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Zadanie 5**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

**Zadanie 6**

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

**Wnioski**

**Przeprowadzone zadania ukazały wszechstronność funkcji w języku C, umożliwiając rozwiązanie problemów zarówno matematycznych, jak i praktycznych. Funkcje ułatwiają zarządzanie kodem i jego ponowne wykorzystanie, co prowadzi do bardziej czytelnych i efektywnych programów.**

**Zadania związane z ciągami liczbowymi pozwoliły na lepsze zrozumienie mechanizmów iteracyjnych i rekurencyjnych, a także na praktyczne zastosowanie tablic do przechowywania danych. Natomiast zadanie dotyczące losowania liczb „Totolotka” pokazało, jak korzystać z funkcji generujących liczby pseudolosowe oraz jak kontrolować przepływ czasu w programie.**

**Funkcje konwertujące jednostki oraz sprawdzające warunki matematyczne, takie jak liczby pitagorejskie, podkreśliły znaczenie precyzyjnego definiowania parametrów i zwracania wyników, co jest kluczowe w programowaniu aplikacyjnym. Dzięki zastosowaniu funkcji kod staje się bardziej modularny, a jego poszczególne elementy łatwiejsze do testowania i debugowania.**