**SPRAWOZDANIE**

|  |  |  |  |
| --- | --- | --- | --- |
| **Przedmiot** | Algorytmy i struktury danych | **Zadanie** | 1\_02 |
| **Autor** | Kamil Borkowski | **Grupa** | WCY22IY1S1 |
| **Temat** | Obliczanie n wyrazów ciągu Fibonacciego | | |

1. Treść

Zaimplementuj algorytm obliczający n wyrazów Ciągu Fibonacciego metodą iteracyjną i rekurencyjną.

* 1. Metoda realizacji

Po wprowadzeniu z klawiatury numeru elementu ciągu, obliczamy jego wartość metodą iteracyjną i rekursyjną.

* 1. Założenia / ograniczenia dotyczące danych:
     1. Dane wejściowe *(opis)*

N-ty element ciągu Fibonacciego – liczba całkowita – wprowadzana z klawiatury

* + 1. Dane wyjściowe *(opis)*

Wartość N-tego elementu ciągu Fibonacciego – liczba całkowita – wprowadzana na ekran

1. Realizacja
   1. Iteracja
      1. Algorytm *(lista kroków lub schemat blokowy)*

a)Deklarujemy zmienne pomocnicze dla pierwszych dwóch elementów ciągu

b)W pętli od i=3 do i <= licznie wprowadzonej z klawiatury obliczamy wartości kolejnych elementów ciągu

c)Po zakończeniu pętli zwracamy wartość n-tego elementu ciągu jako wynik

* + 1. Kod źródłowy

#include <iostream>

using namespace std;

int fibbo\_iteracyjnie(int n)

{

if(n<=2)return 1;

int pom;

int f1=1;

int f2=1;

for(int i=3;i<=n;i++)

{

pom=f1+f2;

f1=f2;

f2=pom;

}

return pom;

}

int main()

{

int n;

cin>>n;

cout<<fibbo\_iteracyjnie(n)<<endl;

return 0;

}

* + 1. Dane wejściowe *(ekran uruchomienia programu)*

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* + 1. Dane wyjściowe *(ekran uruchomienia programu)*

Obraz zawierający zrzut ekranu, tekst, Czcionka

Opis wygenerowany automatycznie

* 1. Rekurencja
     1. Algorytm *(lista kroków lub schemat blokowy)*

a)Jeśli n wynosi 0, zwracana jest wartość 0

b) Jeśli n wynosi 1, zwracana jest wartość 1

c)Jeśli n jest większe od 1 to funkcja wywołuje się rekurencyjnie i zwraca wartość od f(n-1)+f(n-2)

* + 1. Kod źródłowy

#include <iostream>

using namespace std;

int fibbo\_rekurencyjnie(int n)

{

if(n==0)return 0;

if(n==1)return 1;

return fibbo\_rekurencyjnie(n-1)+fibbo\_rekurencyjnie(n-2);

}

int main()

{

int n;

cin>>n;

cout<<fibbo\_rekurencyjnie(n)<<endl;

return 0;

}

* + 1. Dane wejściowe *(ekran uruchomienia programu)*

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

* + 1. Dane wyjściowe *(ekran uruchomienia programu)*

Obraz zawierający zrzut ekranu, tekst, Czcionka

Opis wygenerowany automatycznie

1. Wnioski *(Określ złożoność asymptotyczną algorytmów przy użyciu notacji* ***O lub innej i inne wnioski)***

Dla rozwiązania iteracyjnego: O(n), ponieważ pętla wykona się n razy.

Dla rozwiązania rekursyjnego: O(2^n), ponieważ funkcja wywoła się 2 razy przy każdym wywołaniu