LAB1

1

1. Treść

Zaimplementuj algorytmy obliczające:
wartość n! metodą iteracyjną i rekurencyjną.

1.1. Metoda realizacji

Po wczytaniu wartości n, program oblicza wartość n! na dwa sposoby (iteracyjnie i rekurencyjnie) i wyświetla wyniki na ekranie.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

n – wczytane z klawiatury

1.2.2. Dane wyjściowe

n! – wyświetlone na ekranie

2

1. Treść

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

1.1. Metoda realizacji

Po wczytaniu wartości n, program oblicza wartość n wyrazów Ciągu Fibonacciego na dwa sposoby (iteracyjnie i rekurencyjnie) i wyświetla wyniki na ekranie.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

n – wczytane z klawiatury

1.2.2. Dane wyjściowe

wyniki – wyświetlone na ekranie i w pliku

3

1. Treść

Zaimplementuj algorytm obliczający wartość
wielomianu II stopnia metodami klasyczną i Hornera.

1.1. Metoda realizacji

Po wczytaniu wartości współczynników stojących przy odpowiednich potęgach, program oblicza wartość wielomianu na dwa sposoby (metodą Hornera i klasycznie) i wyświetla wyniki na ekranie.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

a0, a1, a2, x – podawane z klawiatury

1.2.2. Dane wyjściowe

W – wartość wielomianu wyświetlona na ekranie

1. Treść

Zaimplementuj algorytm mnożenia macierzy
o rozmiarach NxM i MxK.

1.1. Metoda realizacji

Po wczytaniu wartości N, M i K, program mnoży wygenerowane macierze i wyświetla macierz wynikową na ekranie.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

N, M, K – podawane z klawiatury

1.2.2. Dane wyjściowe

Macierz wynikowa wyświetlona na ekranie

1. Treść

   Zaimplementuj algorytm wyszukujący maksymalną i minimalną wartość wektora X zawierającego n liczb.

   1.1. Metoda realizacji

   Po wczytaniu wartości n(rozmiaru wektora), program generuje wektor o zadeklarowanej wielkości, wypełnia go liczbami losowymi i wybiera wartości maksymalną i minimalną.

   1.2. Założenia / ograniczenia dotyczące danych:

      1.2.1. Dane wejściowe

      n – ilość liczb losowych, które zawiera wektor

      1.2.2. Dane wyjściowe

      max i min – wyświetlone na ekranie

LAB2

1

1. Treść

   Zaimplementuj realizujący kolejno:
   - Wypełnienie listy kolejnymi wartościami 1 … 100
   - Wyświetlenie zawartości listy
   - Obliczenie i wyświetlenie średniej z wartości umieszczonych w liście
   - Usunięcie z listy wartości nieparzystych
   - Wyświetlenie zawartości listy
   - Obliczenie i wyświetlenie średniej z wartości umieszczonych w liście

   1.1. Metoda realizacji

   Program wypełnia listę wartościami od 1 do 100 i wyświetla utworzoną zawartość, następnie oblicza i wyświetla średnią wartość elementów umieszczonych w liście. Usuwa z listy wartości nieparzyste i ponownie wyświetla jej zawartość. Na koniec ponownie oblicza średnią.

   1.2. Założenia / ograniczenia dotyczące danych:

      1.2.1. Dane wejściowe

      Brak

      1.2.2. Dane wyjściowe

      Wartości zgodnie z opisem działania – wyświetlone na ekranie

2

1. Treść

Zaimplementuj program realizujący w formie listy dwukierunkowej następujące funkcje wizytownika:

- d – dodaj wizytówkę
- s – szukaj wizytówki
- w – wypisz wizytówki
- u – usuń wizytówkę
- z – zapisz rekordy w pliku
- Inne – powtórz sterowanie
- Ctrl-z – kończy działanie

1.1. Metoda realizacji

Stworzenie programu z menu pozwalającym na wykonanie wszystkich funkcji wizytownika określone w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Nazwisko, Imię, nr telefonu – podawane z klawiatury

1.2.2. Dane wyjściowe

Rekordy listy – wyświetlone na ekranie i zapisane do pliku txt

LAB3

1

1. Treść

Zaimplementuj algorytm wyznaczania liczb pierwszych z zadanego przedziału [2,n]. (Użyj listy jednokierunkowej zaimplementowanej wskaźnikowo)

1.1. Metoda realizacji

Wyznaczenie liczb pierwszych z zadanego przedziału przy pomocy Sita Eratostenesa.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

n – wczytane z klawiatury

1.2.2. Dane wyjściowe

Wy.txt – wyświetlone na ekranie i w pliku

2

1. Treść

Zaimplementuj algorytm wyznaczania liczb pierwszych z zadanego przedziału [2,n]. (Użyj listy jednokierunkowej zaimplementowanej wskaźnikowo)

1.1. Metoda realizacji

Wyznaczenie liczb pierwszych z zadanego przedziału przy pomocy Atkina-Bernsteina.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

n – wczytane z klawiatury

1.2.2. Dane wyjściowe

Wy.txt – wyświetlone na ekranie i w pliku

LAB4

1

1. Treść

Konwersja wyrażeń arytmetycznych zapisanych w konwencji infiksowej do wyrażeń zapisanych w konwencji postfisowej przy wykorzystaniu struktury stosu w postaci wskaźnikowej.

1.1. Metoda realizacji

Stworzenie algorytmu opartego na stosie pozwalającego na zamianę wyrażeń arytmetycznych zapisanych w konwencji infiksowej na wyrażenia zapisane w konwencji postfisowej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z klawiatury i we.txt – zawierający wyrażenie zapisane w konwencji infiksowej

1.2.2. Dane wyjściowe

Na ekranie i wy.txt - zawierający wyrażenie zapisane w konwencji postfiksowej

2

1. Treść

Dodawanie dużych liczb przy wykorzystaniu struktury stosu w postaci wskaźnikowej.

1.1. Metoda realizacji

Stworzenie algorytmu opartego na strukturze stosu pozwalającego na dodawanie dużych liczb.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z klawiatury i we.txt – zawierający dodawane liczby

1.2.2. Dane wyjściowe

Na ekranie i wy.txt – zawierający wynik dodawania.

LAB5

1

1. Treść

Zaimplementuj algorytm realizujący rozwiązanie problemu „Wieży Hanoi”

1.1. Metoda realizacji

Stworzenie algorytmu rozwiązującego w kolejnych etapach problem „Wieży Hanoi” w zależności od podanej ilości krążków.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

N – ilość krążków Wieży Hanoi (max 8)

1.2.2. Dane wyjściowe

Ekran i wy.txt – plik zawierający informacje o kolejności przekładania krążków.

2

1. Treść

Zaimplementuj algorytm realizujący symulację łączenia rozmów telefonicznych przez centralę telefoniczną przy założeniach:

- 3 dostępne linie telefoniczne
- 3 rodzaje klientów (priorytety łączenia od najwyższego do najniższego):

P – Klient PLATINUM
G – Klient GOLD
S – Klient SILVER

1.1. Metoda realizacji

Stworzenie programu realizującego symulację centrali telefonicznej przy założeniach zadanych w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

we.txt – plik zawierający informacje o połączeniach przychodzących zawierający (kod klienta, numer klienta, czas trwania rozmowy (w sekundach) separator SPACJA).

1.2.2. Dane wyjściowe

wy.txt – plik zawierający informacje o kolejności uzyskania połączeń przez klientów (abonentów).

Pliki we.txt i wy.txt należy wylistować na ekranie.

LAB6

1

1. Treść

Napisz program, który umożliwi:
Budowanie drzewa BST.
Program powinien:

- Umożliwić wczytanie danych z pliku wejściowego
- Umożliwić wprowadzanie danych z klawiatury
- Dodawanie węzła
- Kasowanie węzła
- Kasowanie danych z całego drzewa
- Narysowanie/wyświetlenie drzewa (BST)
- Wypisanie węzłów drzewa w kolejności przechodzenia
- VLR – pre-order, przejście wzdłużne, prefiksowe
- LVR - in-order, przejście poprzeczne, infiksowe
- LRV – post order, przejście wsteczne, postfiksowe

1.1. Metoda realizacji

Stworzenie programu umożliwiającego stworzenie drzewa BST w oparciu o dane wejściowe z pliku lub klawiatury i obsługującego funkcje zawarte w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z pliku i/lub klawiatury

1.2.2. Dane wyjściowe

Wyświetlane na ekranie

2

1. Treść

Napisz program, który umożliwi:
Budowanie drzewa Kopiec.
Program powinien:

- Umożliwić wczytanie danych z pliku wejściowego
- Umożliwić wprowadzanie danych z klawiatury
- Dodawanie węzła
- Kasowanie węzła
- Kasowanie danych z całego drzewa
- Narysowanie/wyświetlenie drzewa (Kopiec)
- Wypisanie węzłów drzewa w kolejności przechodzenia
- VLR – pre-order, przejście wzdłużne, prefiksowe
- LVR - in-order, przejście poprzeczne, infiksowe
- LRV – post order, przejście wsteczne, postfiksowe

1.1. Metoda realizacji

Stworzenie programu umożliwiającego stworzenie drzewa Kopiec w oparciu o dane wejściowe z pliku lub klawiatury i obsługującego funkcje zawarte w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z pliku i/lub klawiatury

1.2.2. Dane wyjściowe

Wyświetlane na ekranie

LAB7

1

1. Treść

Napisz program, który umożliwi:
Budowanie drzewa AVL.
Program powinien:

- Umożliwić wczytanie danych z pliku wejściowego
- Umożliwić wprowadzanie danych z klawiatury
- Dodawanie węzła
- Kasowanie węzła
- Szukanie węzła (ścieżka)
- Kasowanie danych z całego drzewa
- Narysowanie/wyświetlenie drzewa (AVL)
- Wypisanie węzłów drzewa w kolejności przechodzenia
- VLR – pre-order, przejście wzdłużne, prefiksowe
- LVR - in-order, przejście poprzeczne, infiksowe
- LRV – post order, przejście wsteczne, postfiksowe

1.1. Metoda realizacji

Stworzenie programu umożliwiającego stworzenie drzewa AVL w oparciu o dane wejściowe z pliku lub klawiatury i obsługującego funkcje zawarte w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z pliku i/lub klawiatury

1.2.2. Dane wyjściowe

Wyświetlane na ekranie

1. Treść

Napisz program, który umożliwi:
Budowanie drzewa RB.
Program powinien:

- Umożliwić wczytanie danych z pliku wejściowego
- Umożliwić wprowadzanie danych z klawiatury
- Dodawanie węzła
- Kasowanie węzła
- Szukanie węzła (ścieżka)
- Kasowanie danych z całego drzewa
- Narysowanie/wyświetlenie drzewa (RB)
- Wypisanie węzłów drzewa w kolejności przechodzenia
- VLR – pre-order, przejście wzdłużne, prefiksowe
- LVR - in-order, przejście poprzeczne, infiksowe
- LRV – post order, przejście wsteczne, postfiksowe

1.1. Metoda realizacji

Stworzenie programu umożliwiającego stworzenie drzewa RB w oparciu o dane wejściowe z pliku lub klawiatury i obsługującego funkcje zawarte w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z pliku i/lub klawiatury

1.2.2. Dane wyjściowe

Wyświetlane na ekranie

LAB8

1. Treść

Napisz program, który umożliwi:
Budowanie B-drzewa (stopnia 4).
Program powinien:

- Umożliwić wczytanie danych z pliku wejściowego
- Umożliwić wprowadzanie danych z klawiatury
- Dodawanie węzła
- Kasowanie węzła
- Szukanie węzła (ścieżka)
- Kasowanie danych z całego drzewa
- Narysowanie/wyświetlenie drzewa (B-drzewa)
- Wypisanie węzłów drzewa w kolejności przechodzenia
- LVR - in-order, przejście poprzeczne, infiksowe

1.1. Metoda realizacji

Stworzenie programu umożliwiającego stworzenie drzewa B-drzewa w oparciu o dane wejściowe z pliku lub klawiatury i obsługującego funkcje zawarte w poleceniu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Z pliku i/lub klawiatury

1.2.2. Dane wyjściowe

Wyświetlane na ekranie

LAB9

1

1. Treść

Zaimplementuj algorytm sortowania metodą Shella (ang. Shell Sort).

1.1. Metoda realizacji

Program pyta użytkownika o ilość liczb przeznaczonych do posortowania a następnie prosi o podanie tych liczb. Sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Shell Sort i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z klawiatury.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

2

1. Treść

Zaimplementuj algorytm sortowania metodą biblioteczną (ang. Library Sort). Algorytm bazuje na algorytmie sortowania przez wstawianie, ale z dodawaniem pustych miejsc w tablicy w celu przyspieszenia wstawiania elementów.

1.1. Metoda realizacji

Program pyta użytkownika o ilość liczb przeznaczonych do posortowania a następnie prosi o podanie tych liczb. Sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Library Sort i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z klawiatury.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

3

1. Treść

Zaimplementuj algorytm sortowania metodą szybkiego sortowania (ang. Quick Sort).

1.1. Metoda realizacji

Program pyta użytkownika o ilość liczb przeznaczonych do posortowania a następnie prosi o podanie tych liczb. Sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Quick Sort i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z klawiatury.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

LAB10

1

1. Treść

Zaimplementuj algorytm sortowania metodą stogową (ang. Heap Sort).

1.1. Metoda realizacji

Program pyta użytkownika o ilość liczb przeznaczonych do posortowania a następnie prosi o podanie tych liczb. Sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Heap Sort i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z klawiatury.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

2

1. Treść

Zaimplementuj algorytm sortowania metodą przez zliczanie (ang. Counting Sort).

1.1. Metoda realizacji

Program pyta użytkownika o ilość liczb przeznaczonych do posortowania a następnie prosi o podanie tych liczb. Sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Counting Sort i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z klawiatury.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

3

1. Treść

Zaimplementuj algorytm sortowania metodą przez zliczanie (ang. Comb Sort).

1.1. Metoda realizacji

Program wczytuje liczby przeznaczone do posortowania z pliku a następnie sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Comb Sort i wyświetla je na ekranie w kolejności rosnącej. Możliwe jest także wyświetlenie zawartości pliku przed posortowaniem.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z pliku.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

4

1. Treść

Zaimplementuj algorytm sortowania metodą pozycyjną (ang. Radix Sort).

1.1. Metoda realizacji

Program wczytuje liczby przeznaczone do posortowania z pliku a następnie sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Radix Sort i wyświetla je na ekranie w kolejności rosnącej. Możliwe jest także wyświetlenie zawartości pliku przed posortowaniem.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z pliku.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

LAB11

1

1. Treść

Zaimplementuj algorytm sortowania metodą scalania (ang. Merge Sort).

1.1. Metoda realizacji

Program wczytuje liczby przeznaczone do posortowania z pliku a następnie sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania Merge Sort i wyświetla je na ekranie w kolejności rosnącej. Możliwe jest także wyświetlenie zawartości pliku przed posortowaniem.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z pliku.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

2

1. Treść

Zaimplementuj algorytm sortowania metodą łączenia naturalnego.

1.1. Metoda realizacji

Program wczytuje liczby przeznaczone do posortowania z pliku i wyświetla je na ekranie. Następnie sortuje wczytane liczby całkowite przy pomocy algorytmu sortowania naturalnego i wyświetla je na ekranie w kolejności rosnącej.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Liczby całkowite wprowadzane z pliku.

1.2.2. Dane wyjściowe

Posortowane rosnąco wczytane liczby całkowite.

LAB12

1. Treść

Zaimplementuj algorytm ilustrujący działanie haszowania z próbkowaniem liniowych.

1.1. Metoda realizacji

Stworzenie programu ilustrującego działanie haszowania przy pomocy 10-cio elementowej tablicy i wygenerowaniu 10 losowych łańcuchów znakowych z liter A i B. Jeżeli pojawią się łańcuchy o tej samej wartości funkcji haszującej to próbkowanie liniowe ma umieścić je w innych komórkach, niż wychodzi to z ich haszu.

1.2. Założenia / ograniczenia dotyczące danych:

1.2.1. Dane wejściowe

Losowo wygenerowane czteroznakowe łańcuchy.

1.2.2. Dane wyjściowe

Łańcuch – wartość haszu – liczba obiegów pętli wyszukującej – pozycja łańcucha w tablicy.