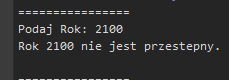
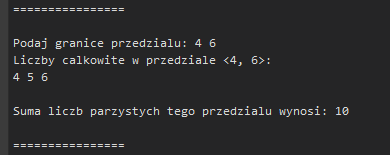
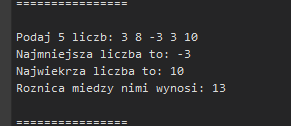
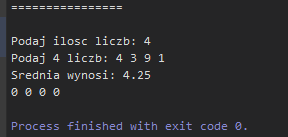
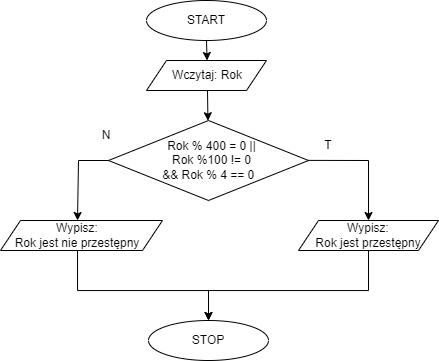
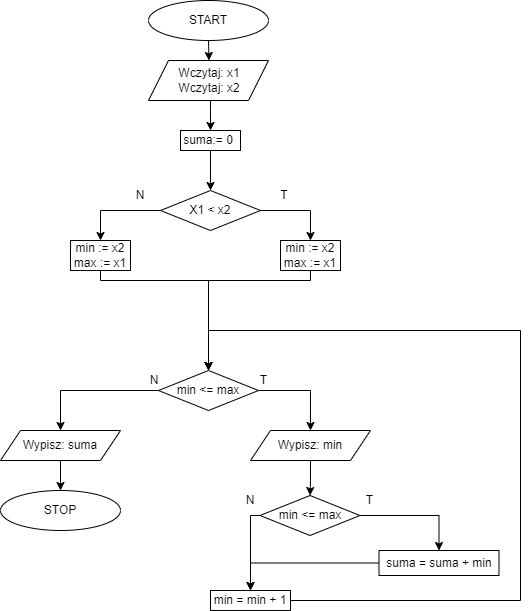
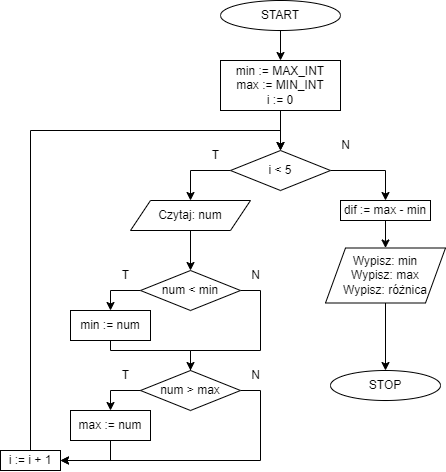
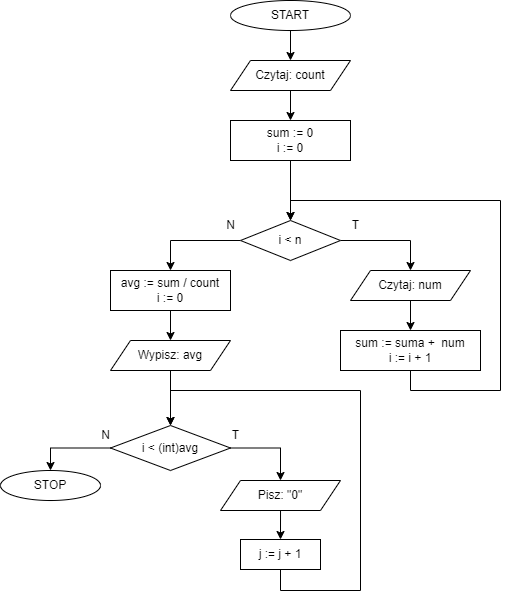
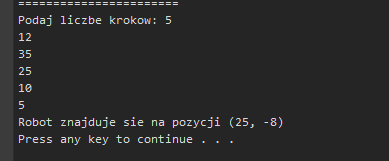
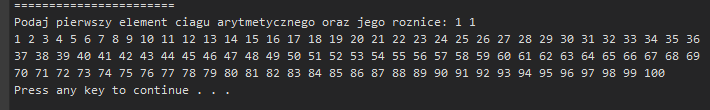
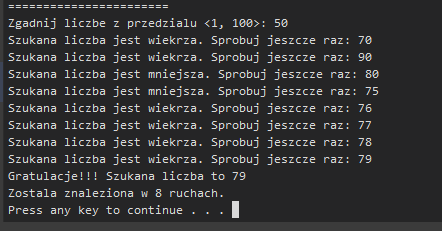
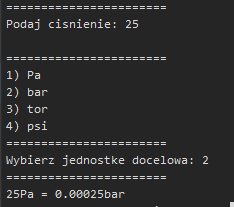
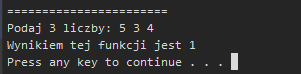
|  |  |  |
| --- | --- | --- |
| **Imię i Nazwisko**  Patryk Twardosz | **Kierunek**  Informatyka Techniczna | **Rok studiów i grupa**  I rok, Gr. 5 |
| **Data zajęć**  Każde | **Numer i temat sprawozdania**  Wszystkie | |

# Instrukcje warunkowe, pętle, schematy blokowe.

1. W celu sprawdzenia czy rok jest przestępny należy sprawdzić jego podzielność przez 4, 100 oraz 400. Jeśli jest podzielne przez 400 lub przez 4, ale nie przez 100 to jest on rokiem przestępnym.
2. Program rozpoczyna wykonywać pętlę, zaczynając od mniejszej liczby, aż do dotarcia do drugiej, większej liczby. Dodatkowo poza wypisywaniem iteratora, jednocześnie jeśli jest on parzysty, do wspólnej zmiennej.  
   
3. Zmienne przechowujące minimalną i maksymalną wartość muszą mieć odpowiednio zainicjalizowaną wartość, jak największą oraz jak najmniejszą, w celu porównywania do podawanych wartości. Podczas wczytywania danych wprowadzonych przez użytkownika, dane są od razu porównywane z obecnie największą i najmniejszą liczbą znalezioną do tej pory. Podkoniec jest wyliczana różnica między największą, a najmniejszą liczbą. 
4. Najpierw program prosi o podanie ilość liczb, z których ma zostać wyliczona średnia, następnie urzytkownik pytany jest o te liczby. Program oblicza średną oraz ją wypisuje, a następnie przy wykorzystaniu pętli for wypisywana jest odpowiednia ilość ‘0’. 
5. Schematu do powyższych zadań:
   * Zad. 1
   * Zad. 2
   * Zad. 3
   * Zad. 4

# Funkcje:

1. Użytkownik podaje liczbę kroków, które ma przejść robocik, a następnie sekwencję odległości, które przeszedł. Program przy pomocy pętli for wczytuje podane wartości oraz wykonuje symulacje kroków w zależności od swojej rotacji. 
2. Po podaniu przez użytkownika pierwszego elementu oraz różnicy ciągu, program wypisze kolejne 100 elementów ciągu oraz zapisze je w tablicy. 
3. Po rozpoczęciu program oblicza pseudo losową liczbę, a następnie w pętli zadaje użytkownikowi pytanie, po wprowadzeniu przez użytkownika liczby, program porównuje ją do wcześniej wylosowanej liczby oraz informuje użytkownika o ich statunku. Po odgadnięciu liczby użytkownik otrzymuje gratulacje. Liczba prób zgadnięcia również jest zliczana i wyświetlana.  
   
4. Program, w odtępach jedno sekundowych, wyświetli w sumie 6 unikalnych liczb z zakresu <1, 49>, symulując tym losowanie w „Totolotku”.  
   
5. Po podaniu przez użytkownika wartości ciśnienia w Pa, program pyta się o jednostkę docelową, po czym konwertuje podaną wartość do żądanej jednostki. 
6. Po podaniu danych a, b oraz c, funkcja sortuje je, po czym dokonuje sprawdzenia poprawności podanych wartości w stosunku Twierdzenia Pitagorasa. 

# Logika:

1. (p ∨ q ⇒ r) ⇒ (p ⇒ r) ∨ (q ⇒ r) - Jest tautologią.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **p** | **q** | **r** | p ∨ q | p ∨ q ⇒ r | p ⇒ r | q ⇒ r | (p ⇒ r) ∨ (q ⇒ r) | (p ∨ q ⇒ r) ⇒ (p ⇒ r) ∨ (q ⇒ r) |
| **0** | **0** | **0** | 0 | 0 | 1 | 1 | 1 | 1 |
| **0** | **0** | **1** | 0 | 1 | 1 | 1 | 1 | 1 |
| **0** | **1** | **0** | 1 | 0 | 1 | 0 | 1 | 1 |
| **0** | **1** | **1** | 1 | 1 | 1 | 1 | 1 | 1 |
| **1** | **0** | **0** | 1 | 0 | 0 | 1 | 1 | 1 |
| **1** | **0** | **1** | 1 | 1 | 1 | 1 | 1 | 1 |
| **1** | **1** | **0** | 1 | 0 | 0 | 0 | 0 | 1 |
| **1** | **1** | **1** | 1 | 1 | 1 | 1 | 1 | 1 |

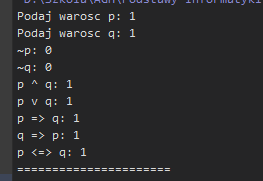
1. (p ∨ q) ∧ (p ⇒ q) ⇒ q ⇒ p - Nie jest tautologią.

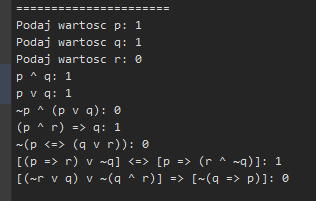
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **p** | **q** | p ∨ q | p ⇒ q | (p ∨ q) ∧ (p ⇒ q) | (p ∨ q) ∧ (p ⇒ q) ⇒ q | (p ∨ q) ∧ (p ⇒ q) ⇒ q ⇒ p |
| **0** | **0** | 0 | 1 | 0 | 1 | 0 |
| **0** | **1** | 1 | 1 | 1 | 1 | 0 |
| **1** | **0** | 1 | 0 | 0 | 1 | 1 |
| **1** | **1** | 1 | 1 | 1 | 1 | 1 |

1. ¬(p ⇒ q) ∧ (q ⇒ p) ⇒ p ∧ ¬q - Jest tautologią.

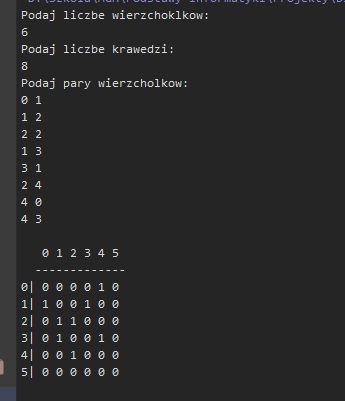
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **p** | **q** | ¬(p ⇒ q) | q ⇒ p | ¬(p ⇒ q) ∧ (q ⇒ p) | p ∧ ¬q | ¬(p ⇒ q) ∧ (q ⇒ p) ⇒ p ∧ ¬q |
| **0** | **0** | 0 | 1 | 0 | 0 | 1 |
| **0** | **1** | 0 | 0 | 0 | 0 | 1 |
| **1** | **0** | 1 | 1 | 1 | 1 | 1 |
| **1** | **1** | 0 | 1 | 0 | 0 | 1 |

1. Program pyta o podanie wartości logicznych dla p oraz q, a następnie wypisuje ich negację, koniunkcję, alternatywę, implikację oraz równoważność.

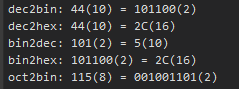


1. p – true q – true r – false
2. p ∧ q - true
3. p ∨ q - true
4. ¬p ∧ (p ∨ q) - false
5. (p ∧ r) → q - true
6. ¬(p ↔(q ∨ r)) - false
7. [(𝑝→𝑟)∨¬𝑞] ↔[𝑝→(𝑟∧¬𝑞)] - true
8. [(¬𝑟∨𝑞)∨¬(𝑞∧𝑟)]→[¬(𝑞→𝑝)] - false

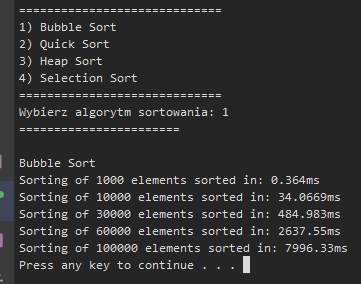
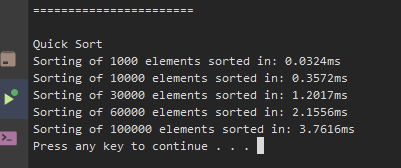
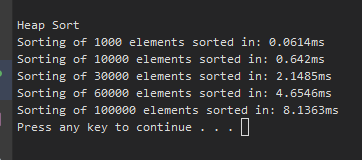
# Struktury danych – Grafy:

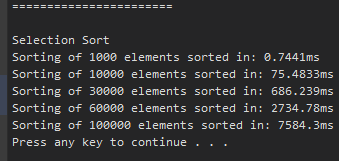
1. Odczytywanie:
   * Liczba wierzchołków: 6  
     Liczba krawędzi: 8
   * Pary:
     + 0 -> 1
     + 1 -> 2
     + 2 -> 2
     + 1 -> 3
     + 3 -> 1
     + 2 -> 4
     + 4 -> 0
     + 4 -> 3
2. Po podaniu danych (ilości wierzchołów I krawędzi oraz połączeń między wierzchołkami) program przedstawi macierz sąsiedztwa w postaci tabeli, gdzie x to wierzchołek początkowy, a y to wierzchołek końcowy.
3. Aby program przedstawiał macierz sąsiedztwa grafu niekierunkowego poza zaznaczeniem sąsiedztwa z *x* do *y*, należy zaznaczyć sąsiedztwo z *y* do *z*.

# Systemy i reprezentacja liczb:

1. 113(10) = 1 \* 2^6 + 1 \* 2^5 + 1 \* 2^4 + 1 \* 2^0 = 1110001(2)  
   432(10) = 1 \* 2^8 + 1 \* 2^7 + 1 \* 2^5 + 1 \* 2^4 = 110110000(2)
2. 555(10) = 2 \* 16^2 + 2 \* 16^1 + 11 \* 16^0 = 22B(16)  
   8736(10) = 2 \* 16^3 + 2 \* 16^2 + 2 \* 16^1 = 2220(16)
3. 1\_0011(2) = 13(16)  
   1\_0100\_1011(2) = 14B(16)
4. D5E7(16) = 13 \* 16^3 + 5 \* 16^2 + 14 \* 16^1 + 7 = 54759(10)  
   F01A33(16) = 15 \* 16^5 + 1 \* 16^3 + 10 \* 16^2 + 3 \* 16^1 + 3 = 15735347(10)
5. 752(8) = 1\_1110\_1010(2) = 1EA(16)   
   2641(8) = 0101\_1010\_0001(2) = 5A1(16)
6. Program konwertuje przykładowe dane wejściowe:  
   

# Złożoność obliczeniowa:

1. Bouble Sort (ms) – O(n2)
2. Quick Sort (ms) – O(n \* log(n))
3. Heap Sort (ms) – O(n \* log(n))

1. Selection Sort (ms) – O(n2)

# Kryptografia:

Program po uruchomieniu wyświetla menu, w którym można wybrać 4 akcje (szyfrowanie i deszyfrowanie w Szyfrze Cezara oraz przy pomocy szyfrowania kluczem XOR.

Po wyborze akcji użytkownik jest proszony o podanie pliku, którego zawartość ma zostać przetworzona oraz o przesunięcie lub maskę, potrzebne do przeprowadzenia odpowiedniej operacji.

