



„POLTELEINFO”
Ogólnopolska Olimpiada Liderów Telekomunikacji i Informatyki
Rok szkolny 2023/2024

Zadania dla grupy informatycznej na zawody II stopnia

Instrukcja dla uczestnika

1. Czas trwania zawodów: 120 minut.
2. II stopień Olimpiady zawiera 5 zadań otwartych.
3. Należy podać poprawną odpowiedź wraz z tokiem rozwiązania.
4. Za każdą prawidłową odpowiedź uzyskuje się maksymalnie 10 punktów. Maksymalna liczba punktów do zdobycia za 5 zadań to 50 punktów.
5. Można korzystać z przyborów do pisania, kalkulatorów i tablic matematycznych oraz rozdawanych kart czystopisu i brudnopisu. Korzystanie z notebooków, tabletów, telefonów komórkowych, smartfonów, smartwatchy, kalkulatorów programowalnych, itp. jest zabronione.

Życzymy powodzenia!

Zadanie 1. Tajemniczy szyfr

1.1. Jak nazywa się szyfr, w którym każdą literę tekstu jawnego zastępuje się inną?

- a) Podstawieniowy.
- b) Asymetryczny.
- c) Przetawieniowy.
- d) Symetryczny.

Odpowiedź: a)

1.2. Jak nazywał się władca, któremu zawdzięczamy jeden z klasycznych szyfrów, w którym każdej literze przypisuje się literę znajdującą się o 3 pozycje dalej w alfabecie?

- a) Juliusz Cezar.
- b) Cesarz Hadrian.
- c) Romulus.
- d) Kaligula.

Odpowiedź: a)

1.3. W szyfrowaniu z użyciem kodu podstawieniowego co oznacza termin "klucz szyfrowania"?

- a) Sposób ustalania długości wiadomości
- b) Unikalny ciąg znaków używany do zaszyfrowania i odszyfrowania
- c) Algorytm przemieszczania liter
- d) Liczba obrotów walca w maszynie szyfrującej Enigmy

Odpowiedź: b)

1.4. Zaznacz zdanie nieprawdziwe o standardowej tablicy ASCII.

- a) Zawiera wszystkie możliwe znaki, które mogą pojawić się w pamięci komputera.
- b) Można ją rozszerzyć np. o polskie znaki.
- c) Pozwala na reprezentację znaków w postaci binarnej.
- d) W tablicy ASCII znak 'a' i 'A' mają dwie różne reprezentacje.

Odpowiedź: a)

1.5. Czy powinniśmy stosować proste szyfry podstawieniowe i przestawieniowe do zabezpieczania istotnych danych? Odpowiedź uzasadnij.

Przykładowa odpowiedź: NIE powinniśmy stosować prostych szyfrów podstawieniowych i przedstawieniowych do zabezpieczania istotnych danych, ponieważ charakteryzują się niskim poziom bezpieczeństwa - są stosunkowo łatwe do złamania i nie zapewniają bezpieczeństwa.

1.6. Poniżej przedstawiono opis słowny oraz przykład pewnego kodu. Przeanalizuj je oraz napisz w pseudokodzie lub wybranym języku programowania dwie funkcje: szyfrującą i deszyfrującą poniższego kodu.

W omawianym kodzie stosuje się alfabet angielski. Każda litera alfabetu angielskiego otrzymuje swój unikatowy numer 0-25 (A – 0, B – 1, C – 2, ..., X – 23, Y – 24, Z – 25). W zakodowanym tekście kolejne litery mają indeksy, które są numerowane od 1. Litera w trakcie kodowania jest podstawiana przez literę znajdującą się o x pozycji dalej w alfabecie, gdzie x oznacza numer pozycji w tekście. Kod jest przewidziany wyłącznie dla wielkich liter. Jeśli podczas przesunięcia nastąpi wyjście poza zakres 25 liczb, należy wrócić do początku alfabetu (patrz ostatnia kolumna przykładu).

Przykład:

| Pozycja | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|----------------------------|----|----|----|----|----|---|----|----|
| Litera do zakodowania | P | R | O | G | R | A | M | Y |
| Numer do zakodowania | 15 | 17 | 14 | 6 | 17 | 0 | 12 | 24 |
| Numer po kodowaniu | 16 | 19 | 17 | 10 | 22 | 6 | 19 | 32 |
| Uwzględnienie zakresu 0-25 | 16 | 19 | 17 | 10 | 22 | 6 | 19 | 6 |
| Litera po kodowaniu | Q | T | R | K | W | G | T | G |

```

import string

letters = list(string.ascii_uppercase)

values_to_letters = {letter: ord(letter) - ord('A') for letter in letters}
letters_to_values = {letter: value for value, letter in values_to_letters.items()}

def encrypt(text):
    encrypted_text = ""
    for index, letter in enumerate(text):
        if letter in values_to_letters.keys():
            position = values_to_letters[letter] + index + 1
            if position > 25:
                position = position - 25
            encrypted_text += letters_to_values.get(position)
    return encrypted_text

def decrypt(text):
    decrypted_text = ""
    for index, letter in enumerate(text):
        if letter in values_to_letters.keys():
            position = values_to_letters[letter] - (index + 1)
            if position > 25:
                position = position - 25
            elif position < 0:
                position = 25 + position
            decrypted_text += letters_to_values.get(position)
    return decrypted_text

```

Zadanie 2.

Poniżej został przedstawiony kod pewnego programu w języku C. Odpowiedz na poniższe pytania.

```

#include <stdio.h>
#include <stdint.h>
unsigned int function(unsigned int a, unsigned int b, unsigned int c,
unsigned int d);
unsigned int my_function(unsigned int a, unsigned int b, unsigned int c,
unsigned int d);
int main()
{
    unsigned int variable_1;
    unsigned int variable_2;
    unsigned int variable_3;

    variable_1 = function(5, 8, 1, 2);
    printf("Pierwszy wynik to: %d\n", variable_1);

    variable_2= my_function(11, 21, 12, 24);
    printf("Drugi wynik to: %d\n", variable_2);

    variable_3 = variable_2<<variable_1;
    printf("Trzeci wynik to: %d\n", variable_3);

}

unsigned int function(unsigned int a, unsigned int b, unsigned int c, un-
signed int d){
    unsigned int wynik_1 = a || b;
    unsigned int wynik_2 = c && d;

    if (wynik_1 == wynik_2) {
        unsigned int result = wynik_1<<wynik_2;
        return result;
    }
    else {
        unsigned int result2 = wynik_1^wynik_2;
        return result2;
    }
}

unsigned int my_function(unsigned int a, unsigned int b, unsigned int c,
unsigned int d){
    unsigned int num_1 = a & b;
    unsigned int num_2 = c | d;
    unsigned int num_3;

    if (num_1>num_2){
        num_3 = num_1+num_2;
    } else if (num_1<num_2){
        num_3 = num_1*num_2;
    }else
        num_3 = num_1*num_1;
    return num_3;
}

```

2.1 Jaki będzie wynik tego programu, co dokładnie wyświetli się użytkownikowi ?

Odpowiedź:

Pierwszy wynik to: 2

Drugi wynik to: 28

Trzeci wynik to: 112

2.2 Opisz działanie funkcji `my_function`

Przykładowa odpowiedź:

Funkcja `my_function` przyjmuje 4 argumenty typu unsigned int. W funkcji `my_function` wykorzystywane są operacje bitowe takie jak: AND oraz OR. Następnie wykorzystywana jest instrukcja if-else, gdzie w zależności od wyrażenia, wykonywane są operacje matematyczne na zmiennych.

2.3 Opisz działanie funkcji `function`

Przykładowa odpowiedź:

W funkcji `function` przyjmuje 4 argumenty typu unsigned int. Zastosowano tu operacje logiczne jak AND i OR. Następnie wykorzystywana jest instrukcja if-else, gdzie w zależności od wyrażenia wykonywane jest przesunięcie bitowe (jeśli wyrażenie w instrukcji if jest prawdziwe) lub w przeciwnym razie jeśli nie wykonywane jest operacja bitowa (XOR).

2.4 Jaki byłby wynik poniższego fragmentu programu, jeśli zamiast tego zapisu

```
variable_3 = variable_2<<variable_1;
printf("Trzeci wynik to: %d\n", variable_3);
```

programista zapisałby kod w taki sposób?

```
variable_3 = variable_2>>variable_1;
printf("Trzeci wynik to: %d\n", variable_3);
```

Odpowiedź:

Trzeci wynik to: 7

2.5 Napisz funkcję w języku C, która zamienia liczbę w systemie dziesiętnym, na liczbę w systemie szesnastkowym.

Przykładowa odpowiedź:

```
#include <stdio.h>

int dec_to_hexa_conversion(int dec_num)
{
    int i = 1, j, temp;
    char hex_num[100];

    while (dec_num != 0) {
        temp = dec_num % 16;
        if (temp < 10)
            temp = temp + 48;
        else
            temp = temp + 55;
    }
}
```

```
        hex_num[i++] = temp;
        dec_num = dec_num / 16;
    }

    printf("Liczba w systemie szesnastkowym to: ");
    for (j = i - 1; j > 0; j--)
        printf("%c", hex_num[j]);
}
```

2.6 Jaki byłby wynik programu, jeśli napisana przez Ciebie funkcja z podpunktu d) przyjęłaby jako parametr poszczególne zmienne: variable_1, variable_2, variable_3.

Odpowiedź

Pierwszy wynik to: 2

Drugi wynik to: 1C

Trzeci wynik to: 70 lub Trzeci wynik to: 7

Zadanie 3.

Poniżej przedstawiono kod w języku Python. Przeanalizuj kod i napisz co wyświetli się w konsoli ?

```

class Fruit():
    def __init__(self, weight, price):
        self.weight = weight
        self.price = price

    def __lt__(self, other):
        return self.weight < other.weight

    def __gt__(self, other):
        return self.price > other.price

    def __str__(self):
        return f'({self.weight}, {self.price})'

    def __repr__(self):
        return str(self)

class Apple(Fruit):
    def __init__(self, variety, weight, price):
        Fruit.__init__(self, weight, price)
        self.variety = variety

    def __str__(self):
        return f'{self.variety}'

l = [Apple('Jonagold', 0.5, 3.4), Apple('Gala', 0.4, 2.5),
      Apple('McIntosh', 0.3, 4.0)]
l.sort()

print(l)
print(sorted(l, key=lambda x: x.variety))
print(sum(map(lambda x: x.price, l)))

```

Odpowiedź:

[McIntosh, Gala, Jonagold]

[Gala, Jonagold, McIntosh]

9.9

Zadanie 4.

Określ najdłuższą wspólną podsekwencję dla słów POLTELEINFO oraz INFORMATYKA. Narysuj i wypełnij siatkę do obliczania najdłuższej wspólnej sekwencji oraz napisz pseudokod dla tego algorytmu.

Podpowiedź: Najdłuższa wspólna podsekwencja to największa liczba liter w sekwencji, które są wspólne dla obu słów.

Przykład:

| | M | O | C |
|---|---|---|---|
| N | 0 | 0 | 0 |
| O | 0 | 1 | 1 |
| C | 0 | 1 | 2 |

Odpowiedź:

- najdłuższa wspólna podskewencja: 4

-siatka

| | P | O | L | T | E | L | E | I | N | F | O |
|---|---|---|---|---|---|---|---|---|---|---|---|
| I | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 2 | 2 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 3 | 3 |
| O | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 3 | 4 |
| R | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 3 | 4 |
| M | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 3 | 4 |
| A | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 3 | 4 |
| T | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| Y | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| K | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| A | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |

lub

| | I | N | F | O | R | M | A | T | Y | K | A |
|---|---|---|---|---|---|---|---|---|---|---|---|
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| O | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| L | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| T | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| E | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| L | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| E | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| I | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| N | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| F | 1 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| O | 1 | 2 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

- pseudokod

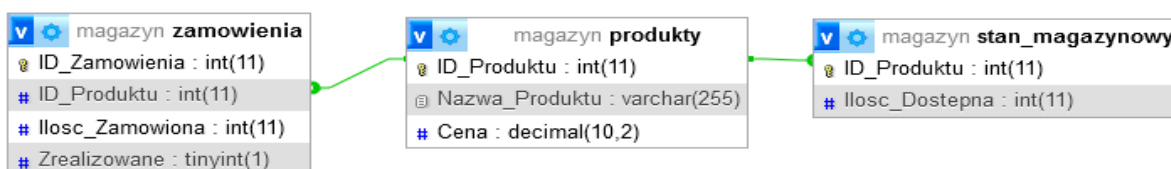
```
if word_a[i] == word_b[j]:
    cell[i][j] = cell[i-1][j-1] + 1
else:
    cell[i][j] = max(cell[i-1][j], cell[i][j-1])
```


Zadanie 5.

Pewien sklep komputerowy prowadzi sprzedaż internetową. Klienci składają zamówienia na produkty dostępne w katalogu (tabela Produkty). W bazie danych zapisywane są dane dotyczące numeru zamówienia, zakupionego produktu, jego ilości oraz status zamówienia. Status zostaje zmieniony przez pracownika gdy ten rozpocznie pakowanie produktów. Niestety w związku z tym następują opóźnienia w aktualizowaniu stanu magazynowego produktów. Pracownicy sklepu chcieliby wiedzieć, które z zamówień są w stanie spakować i wysłać w danej chwili, a do których zamówień muszą domówić towar od dostawców.

Napisz zapytanie SQL, które pokaże produkty, których aktualny stan magazynowy jest równy lub poniżej zamówionej ilości dla wszystkich niezrealizowanych zamówień danego produktu. Pamiętaj aby uwzględnić zrealizowane zamówienia i odjąć spakowane produkty od aktualnego stanu magazynowego.

Dane wejściowe: Baza danych



| Produkty | | |
|-------------|-----------------------|---------|
| ID_Projektu | Nazwa_Projektu | Cena |
| 0 | Monitor | 949.99 |
| 1 | Mysz komputerowa | 299.99 |
| 2 | Procesor | 999.99 |
| 3 | Karta Graficzna | 2099.99 |
| 4 | Pamięć RAM | 450.00 |
| 5 | Zasilacz | 699.99 |
| 6 | Płyta główna | 849.99 |
| 7 | Chłodzenie procesora | 179.99 |
| 8 | Monitor 4K | 1799.99 |
| 9 | Zestaw kabli i spinek | 99.99 |
| 10 | Klawiatura | 399.99 |

| Stan_magazynowy | |
|-----------------|----------------|
| ID_Projektu | Ilosc_dostepna |
| 0 | 10 |
| 1 | 50 |
| 2 | 15 |
| 3 | 4 |
| 4 | 28 |
| 5 | 10 |
| 6 | 8 |
| 7 | 12 |
| 8 | 7 |
| 9 | 80 |
| 10 | 40 |

| Zamowienia | | | |
|---------------|-------------|-----------------|--------------|
| ID_Zamowienia | ID_Projektu | Ilosc_Zamowiona | Zrealizowane |
| 1 | 3 | 5 | 0 |
| 2 | 4 | 4 | 0 |
| 3 | 4 | 2 | 1 |
| 4 | 10 | 5 | 0 |
| 5 | 3 | 1 | 1 |
| 6 | 1 | 10 | 1 |
| 7 | 7 | 5 | 0 |
| 8 | 10 | 2 | 1 |

| | | | |
|----|---|---|---|
| 9 | 7 | 4 | 0 |
| 10 | 5 | 8 | 0 |
| 11 | 7 | 7 | 1 |
| 12 | 2 | 5 | 1 |
| 13 | 8 | 7 | 0 |
| 14 | 8 | 4 | 0 |
| 15 | 2 | 4 | 0 |

Dane wyjściowe: Tabela zawierająca nazwę produktu, cenę, dostępną ilość w magazynie, łączną ilość sztuk z zamówień niezrealizowanych oraz sumę spakowanych sztuk ze zrealizowanych zamówień.

| Nazwa_Produktu | Cena | Ilosc_Dostepna | Ilosc_Zamowiona | Ilosc_Zrealizowana |
|-----------------------------|---------|----------------|-----------------|--------------------|
| Karta Graficzna | 2099.99 | 4 | 5 | 1 |
| Chłodzenie procesora | 179.99 | 12 | 9 | 7 |
| Monitor 4K | 1799.99 | 7 | 11 | NULL |

Przykładowa odpowiedź:

SELECT

P.Nazwa_Produktu,

P.Cena,

SM.Ilosc_Dostepna,

(SELECT SUM(Ilosc_Zamowiona) FROM Zamowienia WHERE ID_Produktu = P.ID_Produktu AND Zrealizowane = 0) AS Ilosc_Zamowiona,

(SELECT SUM(Ilosc_Zamowiona) FROM Zamowienia WHERE ID_Produktu = P.ID_Produktu AND Zrealizowane = 1) AS Ilosc_Zrealizowana

FROM

Produkty P

JOIN

Stan_Magazynowy SM ON P.ID_Produktu = SM.ID_Produktu

JOIN

Zamowienia Z ON P.ID_Produktu = Z.ID_Produktu

WHERE

Z.Zrealizowane = 0 AND

Z.Ilosc_Zamowiona >= SM.Ilosc_Dostepna - COALESCE((SELECT
SUM(Ilosc_Zamowiona) FROM Zamowienia WHERE ID_Projektu = P.ID_Projektu AND
Zrealizowane = 1), 0);