

1. Które z poniższych są instrukcjami pętli w Pythonie?

- for
- while
- repeat
- iterate

**Rozwiązanie:**

Instrukcje for oraz while to pętle (instrukcje sterujące służące do wielokrotnego wykonania innych instrukcji). Instrukcje repeat oraz iterate nie istnieją w języku Python.

2. Rozważmy fragment programu pokazany poniżej:

```
print('x', end='')
for i in range(5, 17, 4):
    print('xx', end='')
print('xxx', end='')
```

Ile znaków x zostanie wypisanych przez powyższy kod?

**Rozwiązanie:**

Program poza pętlą wypisze cztery znaki x. Pętla wykona się trzy razy (dla  $i = 5$ ,  $i = 9$  oraz  $i = 13$ ) wypisując za każdym razem po dwa znaki x. Przy wartości  $i = 17$  pętla zostanie przerwana.

3. Dla jakich zawartości tablicy  $T$  poniższy program wypisze napis poprawiam dokładnie trzy razy?

```
maksimum = 0
for i in range(5):
    if T[i] > maksimum:
        print('poprawiam')
        maksimum = T[i]
```

- [3, 2, 1, 9, 5]
- [1, 2, 3, 2, 1]
- [9, 8, 9, 8, 9]
- [-2, 3, -1, 13, 100]
- [0, 5, 2, 9, 1]

**Rozwiązanie:**

Program przegląda elementy tablicy zgodnie z kolejnością ich występowania i wypisuje poprawiam za każdym razem, gdy liczba z tablicy jest większa niż poprzednie maksimum (początkowo było równe 0).

4. Rozważmy fragment programu pokazany poniżej:

```
p = ???  
s = 'napisik'  
t = ''  
for i in range(len(p)):  
    t = t + s[p[i]]  
print(t)
```

Co należy wstawić w miejsce znaków zapytania, aby program wypisał `pisak`?

- "pisak"
- [ 2, 3, 1, 4, 5 ]
- [ 2, 3, 4, 1, 6 ]
- 23416
- [ 'p', 'i', 's', 'a', 'k' ]

**Rozwiązanie:**

Zmienna `p` musi być tablicą liczb całkowitych (indeksów napisu) a więc tylko druga i trzecia odpowiedź prowadzą do programu, który się interpretuje poprawnie.

Program przegląda kolejne elementy `p` i wypisuje z napisu `s` znaki na tych pozycjach jak element odczytany z `p`.

5. Aby wypisać 17-tą literę alfabetu angielskiego można użyć instrukcji:

- `print('a' + 16)`
- `print('a'.next(16))`
- `print(chr(ord('a') + 16))`

**Rozwiązanie:**

Pierwsza i druga instrukcja są nieprawidłowe. Trzecia instrukcja wypisze znak o kodzie ASCII takim jak kod ASCII litery a plus 16. Litery alfabetu są ustawione w tablicy ASCII po kolei, dlatego będzie to 17-ta litera alfabetu.

6. Dla jakiej wartości zmiennej `x` typu `int` prawdziwy jest warunek  $(x + 1) // 3 * 3 == x$ ?

- 9
- 10
- 19
- 32
- 50
- Dla żadnej z powyższych

**Rozwiązanie:**

Pamiętajmy o tym, że dzielenie zmiennych z użyciem operatora `//` jest w Pythonie całkowitoliczbowe (z zaokrągleniem w dół do najbliższej liczby całkowitej). Powyższy warunek spełniony jest jedynie dla liczb `x` podzielnych przez 3.



7. Ile pamięci operacyjnej zazwyczaj zajmuje zdefiniowana poniżej tablica podczas wykonania programu na komputerze 64-bitowym?

```
tab = []
for i in range(1000):
    tab.append([])
    for j in range(1000):
        tab[i].append(0)
```

- Mniej niż 1 kB
- Około 8 kB
- Około 500 kB
- Około 8 MB
- Więcej niż 1 GB

**Rozwiązanie:**

Zszacowanie zużycia pamięci w Pythonie nie jest takie proste. Jedna zmienna przechowująca liczbę 0 zajmuje 28 bajtów, natomiast tablica jednowymiarowa takich liczb wymiaru 1000 zajmuje około 8000 bajtów. Tysiąc takich tablic zajmuje więc około 8 MB. Do tego należy doliczyć (pomijalnie małą) pamięć zajętą na wskaźniki do tych podtablic.

8. Podciągami słowa  $S$  nazywamy dowolne słowo, które można uzyskać po zakryciu pewnej liczby liter (być może żadnej, być może wszystkich) w słowie  $S$  i odczytaniu pozostałych od lewej do prawej. Podstawem słowa  $S$  nazywamy zaś dowolny jego spójny fragment  $S$  (może być również pusty, może być również całe słowo  $S$ ). Niech  $S = \text{abbaab}$  oraz  $T = \text{babbba}$ . Które z poniższych stwierdzeń są prawdziwe?

- Słowo baab jest podciągami słowa  $S$ , ale nie jest podciągami słowa  $T$ .
- Słowo aaa jest podstawem słowa  $S$ .
- Słowo abb jest podstawem słów  $S$  oraz  $T$ .
- Słowo bab jest podciągami słów  $S$  oraz  $T$ .
- Słowo bbbb jest podstawem słowa  $T$ .

**Rozwiązanie:**

Główną trudnością zadania było jedynie zrozumienie treści i definicji podciągu (który nie musi być spójny) oraz podstawu (które musi być spójne).

9. Rozważmy słowa o długości 5 złożone jedynie z liter a oraz b. Ile spośród tych słów nie zawiera trzech sąsiednich jednakowych liter?

16

**Rozwiązanie:**

Są 32 pięcioliterowe słowa składające się z liter a oraz b. Obliczmy ile jest tych, które zawierają trzy sąsiednie jednakowe litery (czyli ile jest „złych” słów). Są to słowa postaci aaa??, bbb??, abbb?, baaa?, ?abbb, ?baaa, gdzie ? oznacza dowolną literę (a lub b). Każde złe słowo zostało w ten sposób policzone dokładnie raz. Razem jest 16 złych słów, a więc  $32 - 16 = 16$  spełniających warunki zadania.



10. Rozważmy fragment programu pokazany poniżej:

```
liczba = ???
while liczba > 0:
    if (liczba > 5) and (liczba < 95):
        liczba -= 10
    liczba = max(0, liczba)
    liczba = min(100, liczba)
```

Dla których z poniższych wartości wpisanych zamiast znaków zapytania program zapętlę się (tzn. nie zakończy swojego działania)?

- 3
- 2
- 28
- 33
- 95
- 105

**Rozwiązanie:**

Jeśli  $liczba \leq 0$  to pętla nie wykonuje się wcale, a więc program się nie zapętlę. Wewnątrz pętli jeżeli w którymkolwiek momencie wartość zmiennej `liczba` nie będzie w zakresie  $[6, 94]$  to program się zapętlę. Dwie ostatnie instrukcje dbają o to, aby wartość zmiennej cały czas była w przedziale  $[0, 100]$ . Jeśli zatem pierwsza iteracja pętli będzie z wartością `liczba` większą niż 94 lub mniejszą niż 6 to program się zapętlę. W pozostałych zaś przypadkach, jeśli wartość zmiennej `liczba` będzie dawać resztę 0, 5, 6, 7, 8 lub 9 przy dzieleniu przez 10 to program zakończy swoje działanie, bo każda iteracja pętli będzie zmniejszała wartość zmiennej o 10 aż wartość spadnie do mniejszej lub równej 0, co zakończy pętlę. Jeśli jednak reszta z dzielenia przez 10 będzie równa 1, 2, 3 lub 4 to program się zapętlę (kilka pierwszych iteracji zredukuje wartość zmiennej do 1, 2, 3 lub 4).

11. Które z poniższych liczb zapisanych w systemie szesnastkowym są większe niż liczba 100 zapisana w systemie dziesiętkowym?

- $AF_{16}$
- $50_{16}$
- $64_{16}$
- $F00_{16}$
- $C0_{16}$

**Rozwiązanie:**

Liczba  $100 = 6 \cdot 16 + 4$ , a zatem  $100_{10} = 64_{16}$ . A zatem jedynie  $50_{16} = 80_{10}$  oraz  $64_{16} = 100_{10}$  nie są większe niż 100.

12. Rozważmy poniższą funkcję:

```
def funkcja(liczby):
    wynik = []
    ile = {}
    for x in liczby:
        if x not in ile:
            ile[x] = 0
        ile[x] += 1
    for element in sorted(ile.items()):
        for i in range(element[1]):
            wynik.append(element[0])
    return wynik
```

Co zwraca powyższa funkcja?

- Liczbę wystąpień każdego elementu ciągu
- Posortowany ciąg
- Ciąg z usuniętymi powtarzającymi się elementami
- Ciąg, w którym każdy element występuje tyle razy ile wynosi jego wartość

**Rozwiązanie:**

Przedstawiona funkcja wykonuje sortowanie przez zliczanie: najpierw tworzy kolekcję par (element, liczba wystąpień elementu) z użyciem słownika (dict). Funkcja następnie przegląda kolejne pary w kolejności zgodnej z posortowaniem (rosnąco względem elementu ciągu) i w strukturze wynikowej umieszcza tyle kopii tego elementu ile zostało zliczone w ciągu wejściowym.

13. Rozważmy fragment programu pokazany poniżej:

```
def obliczaj(n):
    if n == 10: return n
    return 1 + obliczaj(n + 2)
```

Jaki wynik zwróci wywołanie obliczaj(4)?

- 3
- 6
- 10
- 13
- 16
- program nie zakończy w ogóle działania (zapętlę się)

**Rozwiązanie:**

Wywołanie obliczaj(4) zwraca 1 + obliczaj(6), czyli 2 + obliczaj(8), czyli 3 + obliczaj(10) = 3 + 10 = 13.

14. **Palindromem nazywamy słowo, które czytane od lewej brzmi tak samo jak czytane od prawej (na przykład anna lub kajak).**

**Ile najmniej liter należy zamienić na inne w słowie abaicka, żeby uzyskać palindrom?**

**Rozwiązanie:**

Wystarczy zamienić na przykład literę drugą i trzecią, aby uzyskać akcicka, które jest palindromem. Zmieniając jedną literę można usatysfakcjonować tylko jedną parę znaków  $i$ -ty oraz  $(8 - i)$ -ty, a zatem nie da się uzyskać palindromu zmieniając mniej liter.

15. **Rozważmy fragment programu pokazany poniżej:**

```
x = ???
if x % 3 == 0:
    print('wak', end='')
    i = 0
    while i+10 < x:
        print('tat', end='')
        i += 1
    print('acje', end='')
```

Jaką największą liczbę naturalną można wstawić w miejsce znaków zapytania, aby program wypisał napis wakacje?

**Rozwiązanie:**

Aby wypisany napis był wakacje, pętla nie może wykonać się ani razu. A zatem nie może zachodzić warunek  $0 + 10 < x$ . Czyli  $x \leq 10$ . Oprócz tego, aby cokolwiek w ogóle się wypisało, liczba  $x$  musi być podzielna przez 3. Największą liczbą podzielną przez 3 i jednocześnie nie przekraczającą 10 jest liczba 9.

16. **Palindromem nazywamy słowo, które czytane od lewej brzmi tak samo jak czytane od prawej. Alfabet angielski składa się z 26 liter.**

Ile jest palindromów wśród wszystkich słów długości 4 składających się jedynie z liter występujących w alfabecie angielskim?

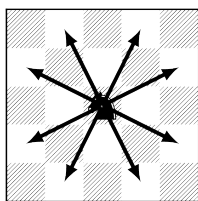
**Rozwiązanie:**

Pierwsze dwie litery można ustawić dowolnie (na jeden z  $26 \cdot 26 = 676$  sposobów. To też jednoznacznie determinuje jakie powinny być dwie ostatnie litery.

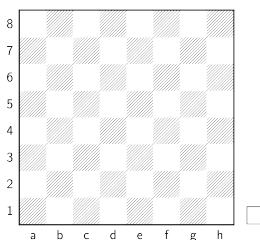
17. **Na ile sposobów możemy zapisać liczbę 5 jako sumę co najmniej dwóch dodatnich składników naturalnych? Sposoby różniące się jedynie kolejnością składników uznajemy za jednakowe.****Rozwiązanie:**

$5 = 4 + 1 = 3 + 2 = 3 + 1 + 1 = 2 + 2 + 1 = 2 + 1 + 1 + 1 = 1 + 1 + 1 + 1 + 1$ .

18. Skoczek szachowy porusza się w jednym z ośmiu kierunków zgodnie z rysunkiem po lewej stronie. Szachownica wygląda jak na rysunku po prawej stronie.



Ruchy skoczka szachowego



Pola szachownicy

Jaka jest najmniejsza dodatnia liczba ruchów, które może wykonać skoczek, aby startując z pola a1 mógł powrócić w położenie początkowe? Zakładamy, że skoczek nie może zawracać, tzn. wykonując następny ruch nie może skoczyć na pole, na którym był bezpośrednio przed wykonaniem ostatniego ruchu.

4

**Rozwiązanie:**

Każdy skok zmienia kolor pola, więc wynik musi być parzysty. Ponadto, warunki zadania zabraniają zawracać, więc wynik musi być równy co najmniej 4. Sposób jego osiągnięcia jest na przykład taki: a1 → b3 → d4 → c2 → a1.

19. Dzielnikiem liczby naturalnej  $n$  jest każda liczba naturalna  $m$  taka, że  $\frac{n}{m}$  jest liczbą całkowitą. Ile dzielników ma liczba 240?

20

**Rozwiązanie:**

Rozwińmy to zadanie na dwa sposoby:

(Sposób 1) Jeśli liczba  $n$  ma dzielnik  $m$  to ma również dzielnik  $\frac{n}{m}$ . Podzielmy dzielniki na „małe” (mniejsze od  $\sqrt{n}$ ) oraz duże (większe od  $\sqrt{n}$ ). Każdemu małemu dzielnikowi odpowiada unikalny duży dzielnik. Dla liczb będących kwadratami liczby naturalnej mamy jeszcze dzielnik  $\sqrt{n}$ . Liczba 240 nie jest kwadratem liczby naturalnej i ma następujące małe dzielniki 1, 2, 3, 4, 5, 6, 8, 10, 12, 15 (10 dzielników). A zatem jest też 10 dużych dzielników: 240, 120, 80, 60, 48, 40, 32, 24, 20 oraz 16, czyli w sumie 20 dzielników.

(Sposób 2) Rozkładamy liczbę 240 na czynniki pierwsze:  $240 = 2^4 \cdot 3 \cdot 5$ . Każdy dzielnik liczby 240 musi więc być postaci:  $2^\alpha \cdot 3^\beta \cdot 5^\gamma$ , gdzie  $\alpha \in \{0, 1, 2, 3, 4\}$ ,  $\beta \in \{0, 1\}$ ,  $\gamma \in \{0, 1\}$ . Ponieważ każda liczba ma unikalny rozkład na czynniki pierwsze oraz każdy rozkład na czynniki pierwsze definiuje unikalną liczbę, jest  $5 \cdot 2 \cdot 2 = 20$  dzielników.

20. Rozważmy fragment programu pokazany poniżej:

```
def czy1(n):
    if n == 0: return False
    return (n % 10 == 1) or (czy1(n // 10))

...

ile1 = 0
for i in range(1, 254):
    if czy1(i):
        ile1 += 1
print(ile1)

...
```

Jaką liczbę wypisze powyższy program?

134

**Rozwiązanie:**

Funkcja czy1 zwraca True wtedy i tylko wtedy, gdy liczba  $n$  ma jedynekę w zapisie dziesiętnym.

Kod zaś zlicza ile spośród liczb od 1 do 253 włącznie mają w swoim zapisie dziesiętnym chociaż jedną jedynekę. Są to liczby: 1 (jedna liczba), 10, 11...19 (10 liczb), 21, 31, ..., 91 (8 liczb), 100, 101...199 (100 liczb), 201 (jedna liczba), 210, 211, ..., 219 (10 liczb), 221, 231, 241, 251 (cztery liczby).

21. Celem poniższej funkcji jest obliczenie ile jest w ciągu (liczb naturalnych) maksimów lokalnych, czyli takich elementów, które są większe od obu swoich sąsiadów. Jeśli element jest pierwszy lub ostatni w ciągu, wystarczy żeby był większy od swojego jedynego sąsiada.

```
def ile_maks_lokalnych(liczby):
    ile, poprzednia, jeszcze_wczesniejsza = 0, -1, -1
    liczby.append(-1)
    for liczba in liczby:
        if (poprzednia > jeszcze_wczesniejsza) and (poprzednia > liczba):
            ile += 1
            ???
        poprzednia = liczba
    liczby.pop()
    return ile
```

Jaką linię należy wstawić w miejsce znaków zapytania?

- Nic nie trzeba wpisać (wystarczy zmazać znaki zapytania)
- liczba = poprzednia
- jeszcze\_wczesniejsza = liczba
- jeszcze\_wczesniejsza = poprzednia
- jeszcze\_wczesniejsza = max(liczba, poprzednia)
- poprzednia = jeszcze\_wczesniejsza



**Rozwiązanie:**

Funkcja przeglądając kolejne liczby w ciągu cały czas utrzymuje zmienne poprzednia oraz jeszcze\_wczesniejsza. W momencie, gdy zmienna liczba zawiera  $i$ -tą liczbę, zmienne te przechowują odpowiednio  $(i - 1)$ -szą oraz  $(i - 2)$ -gą liczbę ciągu. Zakładamy przy tym, że ciąg dopełniony jest z lewej i prawej wartościami  $-1$  czyli mniejszymi niż wszystko co mogło wcześniej być w strukturze liczby (element  $-1$  na końcu jest nawet rzeczywiście tymczasowo dodawany do struktury, a po wykonaniu obliczeń usuwany).

Po sprawdzeniu warunku na maksimum lokalne konieczne jest przygotowanie się do następnej iteracji (do zwiększenia  $i$ ). Liczba jeszcze\_wczesniejsza powinna zostać zatracona. Liczba poprzednia powinna zostać przepisana do zmiennej jeszcze\_wczesniejsza (to jest odpowiedź zadania), zaś liczba powinna zostać przepisana do zmiennej poprzednia.

**22. Rozważmy fragment programu pokazany poniżej:**

```
elementy = [(3, 2), (2, 5), (2, 9), (1, 7), (9, 0)]
elementy.sort()
print(elementy[2][0], elementy[2][1])
```

Co wypisze powyższy kod?

- 3 2
- 2 5
- 2 9
- 1 7
- 9 0

**Rozwiązanie:**

Tuple sortowane są leksykograficznie: tzn. najpierw względem pierwszego elementu, a w przypadku remisu uwzględniane są kolejne elementy aż do znalezienia różnicy.

A zatem kolejność po posortowaniu będzie: (1, 7), (2, 5), (2, 9), (3, 2), (9, 0).

**23. Dla jakich wartości parametru liczba poniższa funkcja zwraca 17?**

```
def funkcja(liczba):
    if liczba == 0:
        return 1
    return funkcja(liczba // 2) + liczba % 2
```

- 17
- 34
- 289
- $2^{17}$
- $255 \cdot 257$

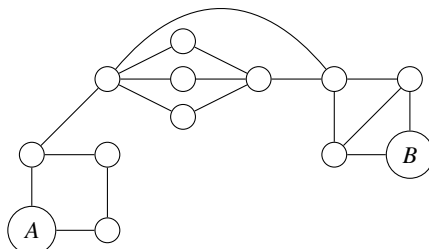
**Rozwiązanie:**

Funkcja zwraca 1 plus liczbę zapalonych bitów w zapisie dwójkowym liczby podanej jako parametr. Aby więc funkcja mogła zwrócić 17, musi otrzymać liczbę, która ma szesnaście jedynek w zapisie dwójkowym co dyskwalifikuje liczby 17, 34 oraz 289 (mają one co najwyżej 9 bitów łącznie). Liczba  $2^{17}$  ma tylko jedną jedynkę w zapisie dwójkowym (i siedemnaście zer po niej).

Aby przekonać się, że liczba  $255 \cdot 257$  jest rozwiązaniem zadania, wystarczy skorzystać z wzoru skróconego mnożenia:  $255 \cdot 257 = (256 - 1)(256 + 1) = (2^8 - 1)(2^8 + 1) = 2^{16} - 1$ . Liczba ta ma w zapisie dwójkowym dokładnie szesnaście jedynek (i nic więcej).



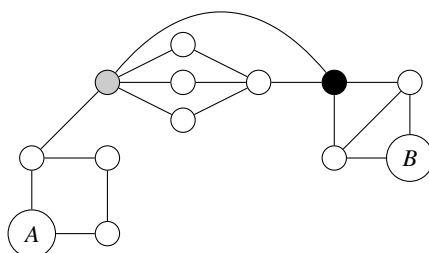
24. Ile jest ścieżek siedmiokrawędziowych między wierzchołkami  $A$  i  $B$  w poniższym grafie? W tym zadaniu rozpatrujemy jedynie ścieżki, w których wszystkie odwiedzone wierzchołki muszą być parami różne. Innymi słowy, ile jest na poniższym rysunku ścieżek, które przechodzą od punktu  $A$  do punktu  $B$  poruszając się od punktu do punktu po narysowanych odcinkach, odwiedzając dokładnie 7 takich odcinków i nie przechodząc przez żaden punkt dwukrotnie?



8

**Rozwiązanie:**

Skupmy się na zliczeniu ścieżek długości 4 oraz 5 od  $A$  do zaczernionego wierzchołka. Wynikiem zadania będzie liczba ścieżek długości 4 plus dwukrotność liczby ścieżek długości 5.



Z wierzchołka  $A$  do szarego wierzchołka można dojść jedynie na dwa sposoby: ścieżką długości 2 lub długości 4. Z szarego wierzchołka jest jedna bezpośrednia krawędź do zaczernionego wierzchołka oraz trzy ścieżki trzykrawędziowe.  $A$  zatem do zaczernionego wierzchołka nie ma żadnej ścieżki długości 4 oraz są cztery ścieżki długości 5. To oznacza że siedmiokrawędziowych ścieżek między  $A$  i  $B$  jest  $0 + 2 \cdot 4 = 8$ .

25. Rozważmy fragment programu pokazany poniżej:

```
ile = 0
liczby = [5, 3, 2, 9, 5, 2, 9, 3, 1, 7]
for i in range(len(liczby)):
    for j in range(i+1, len(liczby)):
        if (liczby[i] + liczby[j]) % 2 == 0:
            ile += 1
print(ile)
```

Jaką liczbę wypisze na wyjście ten program?

29

**Rozwiązanie:**

Program zlicza liczbę par  $(i, j)$  ( $i < j$ ) pozycji w ciągu, których suma odpowiadających im wartości jest parzysta. W ciągu są dwie liczby parzyste oraz osiem liczb nieparzystych. Aby uzyskać sumę parzystą należy dodać do siebie albo dwie liczby parzyste (tylko 1 sposób), albo dwie liczby nieparzyste ( $\frac{8 \cdot 7}{2} = 28$  sposobów). Razem 29 par indeksów  $(i, j)$ , które spowodują zwiększenie zmiennej ile.

26. **Rozważmy fragment programu pokazany poniżej:**

```
i, liczba = 1, ???
while liczba <= 10:
    liczba = liczba + 1
    i += 2
}
print(i)
```

Program po uruchomieniu wypisał na ekranie liczbę 11. Jaką najmniejszą liczbę całkowitą można wstawić w miejsce znaków zapytania, aby tak się stało?

**Rozwiązanie:**

Każdy obrót pętli for powoduje zwiększenie zmiennej  $i$  o 2. Pętla startuje z  $i = 1$ , a zatem musi wykonać się dokładnie  $\frac{11-1}{2} = 5$  iteracji pętli.

Instrukcja `liczba = (2 * liczba + 3) // 2` jest równoważna zwiększeniu zmiennej `liczba` o 1. Pętla ma się przerwać co pięciu iteracjach na wartości zmiennej `liczba` równej 11, więc jej wartość początkowa musiała być równa 6.

27. **Rozważmy poniższą funkcję:**

```
def funkcja(liczba):
    if liczba == 1: return 0
    return 1 + funkcja(liczba // 2)
```

Jaką wartość zwróci wywołanie `funkcja(1000000)`?

**Rozwiązanie:**

Funkcja oblicza logarytm dwójkowy z liczby zaokrąglony w dół do najbliższej liczby całkowitej.

Ponieważ  $2^{20} = 2^{10} \cdot 2^{10} = 1024 \cdot 1024 > 1000 \cdot 1000 = 1\,000\,000$  to wynik zadania jest co najwyżej równy 19. Z drugiej strony  $2^{19} = 2^{10} \cdot 2^9 = 1024 \cdot 512 < 1100 \cdot 600 = 660\,000$ , więc wynik zadania jest co najmniej równy 19.

28. **Rozważmy program, który wczytuje liczby naturalne  $n$  oraz  $m$  i wykonuje potem dokładnie  $2^n \cdot m$  operacji elementarnych w celu obliczenia wyniku. Uruchamiono program na komputerze, który może wykonać  $10^8$  operacji elementarnych w ciągu sekundy. Dla jakiej wartości  $n$  i  $m$  wykonanie programu zajmie co najwyżej dwie sekundy?**

- $n = 10, m = 10$
- $n = 100, m = 100$
- $n = 10, m = 100$
- $n = 100, m = 10$
- $n = 1, m = 1\,000\,000$

**Rozwiązanie:**

Ponieważ  $2^{100} = (2^{10})^{10} = 1024^{10} > 1000^{10} = 10^{30}$ , eliminuje to od razu z rozważań odpowiedzi z  $n = 100$ .  $2^{10}$  to tylko 1024, więc nawet przemnożone przez 100 mieści się w limicie  $10^8$  operacji, co oznacza, że należy zaznaczyć odpowiedź pierwszą i trzecią.  
Ostatnią odpowiedź również należy zaznaczyć, bo:  $2 \cdot 1\,000\,000 = 2\,000\,000 < 10^8$ .

29. **Rozważmy fragment programu pokazany poniżej:**

```
ile = 0
liczby = [4, 9, 8, 5, 3, 9, 3, 0, 10, 3]
for i in range(len(liczby)):
    for j in range(i+1, len(liczby)):
        if liczby[i] + liczby[j] >= 10:
            ile += 1
print(ile)
```

**Jaką liczbę wypisze na wyjście ten program?****Rozwiązanie:**

Powyższy program oblicza liczbę par indeksów  $(i, j)$  ( $i < j$ ) w ciągu, których odpowiadające wartości sumują się do wartości co najmniej 10.  
Odpowiedź zadania nie zmieni się, jeśli posortujemy cały ciąg: (0, 3, 3, 3, 4, 5, 8, 9, 9, 10). Liczba 0 utworzy szukaną parę jedynie z liczbą 10. Liczby 3, 3, 3, 4 oraz 5 utworzą pary jedynie z jedną z liczb 8, 9, 9 lub 10 ( $5 \cdot 4 = 20$  par). Liczby 8, 9, 9, 10 tworzą też pary (każda z każdą) między sobą. Jest 6 takich par. Wynik zadania to zatem  $1 + 20 + 6 = 27$ .

30. **Pewien program wczytuje ciąg liczb o długości  $n$  do tablicy, następnie sortuje go używając funkcji `std::sort` i wypisuje wszystkie elementy zgodnie z posortowaniem. Jaką złożoność obliczeniową ma ten program?**

- $\Theta(\log n)$
- $\Theta(\sqrt{n})$
- $\Theta(n)$
- $\Theta(n \log n)$
- $\Theta(n^2)$

**Rozwiązanie:**

Dominujący czas programu to w tym przypadku sortowanie, które zajmuje  $\Theta(n \log n)$ .