

Za podanie odpowiedzi brawa dla:

- Morasiu
- Dawida Burnat
- [tu możesz być ty!]
- Katarzyny Twardowskiej (MrocznyNietoperzBatman124)

Zapraszam do dodawania pytań na Discord i odpowiedzi w niniejszym arkuszu~!

<https://e.wsei.edu.pl/mod/quiz/view.php?id=5604>
cs201-stac

Do wpisania

Jaka będzie wartość zmiennej

Jaka będzie wartość zmiennej `k` po wykonaniu podanego fragmentu kodu

```
01. int k = 1;
02. int i = 2;
03. while( i < 10 )
04. {
05.     if( i % 2 == 0 )
06.         k = k++;
07.     i++;
08. }
```

Wpisz wartość.

Odpowiedź:

Odpowiedź: 4

Ile razy wykona się

Ile razy wykona się instrukcja zawarta w podanej pętli?

Program zapisany jest w pseudokodzie.

```
x ← 1
do
  instrukcja
  x ← x + 4
while (x ≤ 7)
```

Podaj liczbę wywołań instrukcji

Odpowiedź: 2

Ustal, co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 5, 6, 7 podanych kolejno:

```
int fun(int n) {
  if (n < 2) return n;
  if (n % 2 == 0) return fun(n-3) + 1;
  else return fun(n + 1)+1;
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).

Jeśli funkcja się zapętlą w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź:

Odpowiedź: 5;4;7

Ustal, co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 6, 7, 8 podanych kolejno:

```
int fun(int n) {  
    if (n < 2) return n;  
    else return fun(n - 1);  
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).
Jeśli funkcja się zapętla w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź: 1,1,1,

Podaj w systemie

Podaj, w systemie dziesiętkowym, wartość liczby zakodowanej w systemie **uzupełnienie-do-dwa**:

$$(00001110)_{u2} = (\dots)_{10}$$

Odpowiedź:

Odpowiedź: 14

<http://www.kalkmat.pl/U2/Decimal>

Podaj wyraz zakodowany w UTF-8

Podaj wyraz zakodowany w UTF-8 (bez BOM). Wielkość liter jest istotna .

7772C3B3C5BC6B61

Odpowiedź:

Odpowiedź: wróżka(konwertując do ASCII w notepadzie++)

<https://sites.google.com/site/nathanlexwww/tools/utf8-convert>

hexa

Jeśli jesteś nowym klientem banku

Jeśli jesteś nowym klientem otwierającym konto z kartą kredytową, otrzymujesz 15% zniżkę na dzisiejsze zakupy. Jeśli jesteś stałym klientem banku i masz kartę lojalnościową, otrzymujesz 10% zniżkę. Jeśli masz kupon polecający, otrzymujesz 20% zniżki na dzisiejsze zakupy, ale kupon nie może być wykorzystany ze zniżką dla nowego klienta. Zniżki się sumują (jeśli to nie jest wykluczone powyższym regulaminem).

Uzupełnij tablicę decyzyjną (przeciągnij i upuść markery na odpowiednie pola):

Warunki	Reguła 1	Reguła 2	Reguła 3	Reguła 4	Reguła 5	Reguła 6
Nowy klient (15%)	Tak	3	6	9	Nie	14
Karta lojalnościowa (10%)	1	4	7	10	12	15
Kupon (20%)	2	5	8	11	13	16
Działanie						
Zniżka (%)	20	15	30	10	20	0

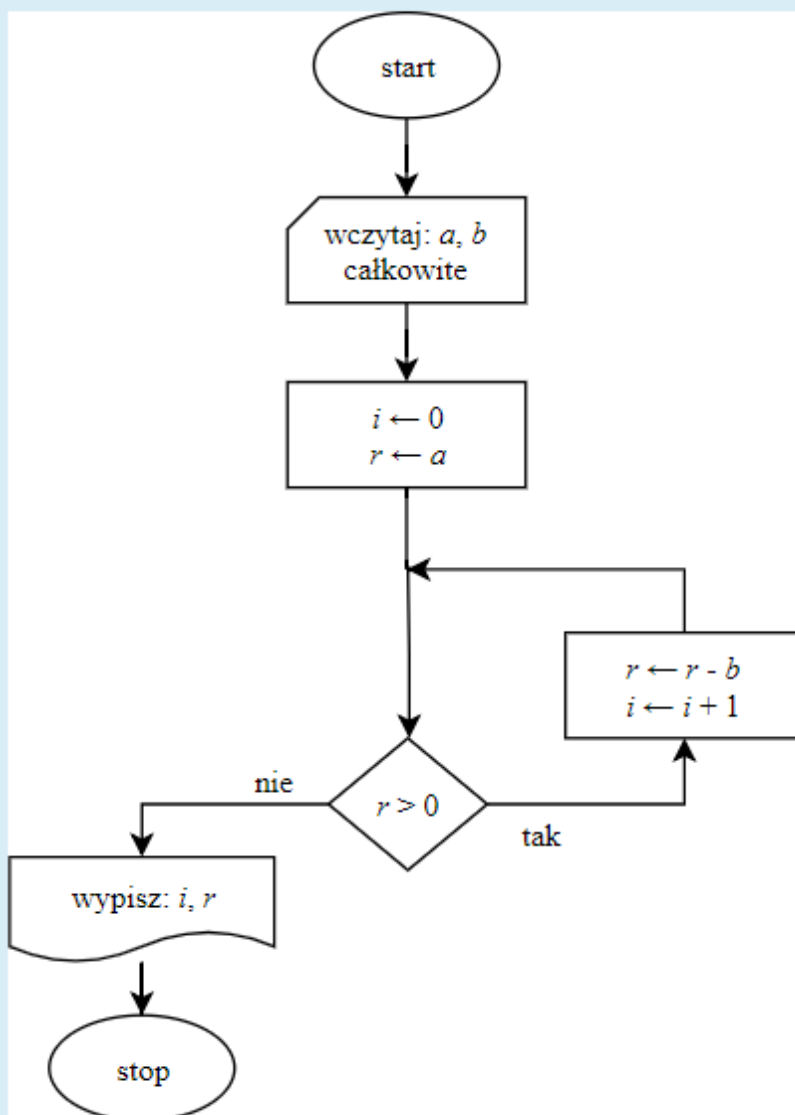
TakNie--

Odpowiedź:

Reguła 1	Reguła 2	Reguła 3	Reguła 4	Reguła 5	Reguła 6
Tak				Nie	

Algorytm jaka będzie wartość zmiennej i

Jaka będzie wartość zmiennej i po zakończeniu działania algorytmu, jeśli na wejściu wczytano wartości całkowite $a = 23$ oraz $b = -8$.

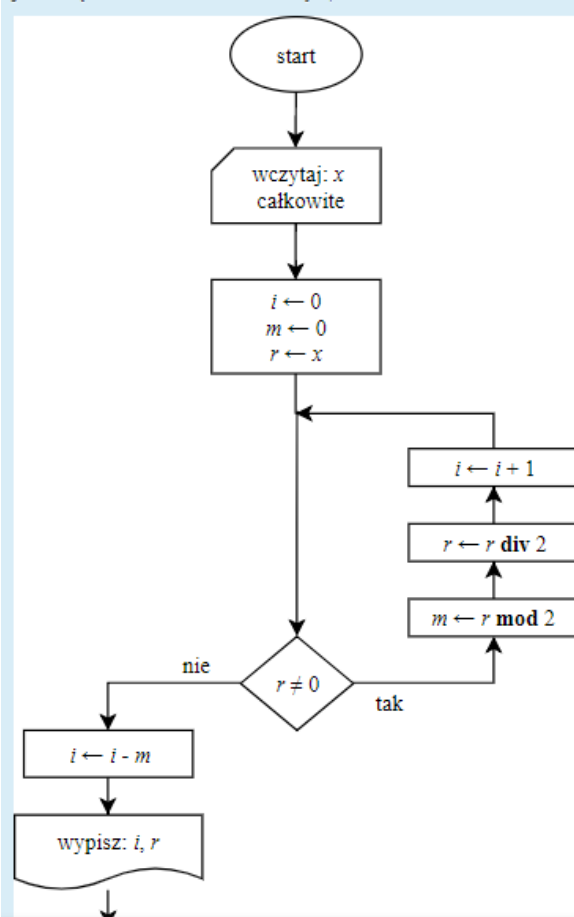


Podaj wyznaczoną wartość lub jeśli algorytm się zapętla.

Odpowiedź:

Odpowiedź: -9999

Jaka będzie wartość zmiennej i po zakończeniu działania algorytmu, jeśli na wejściu wczytano wartość całkowitą $x = -1$.



Ta strona używa ciasteczek (cookies), dzięki którym nasz serwis może działać lepiej.

$i=2, r=0$

Zaznacz kilka

Zaznacz słowa zarezerwowane

Zaznacz słowa zarezerwowane (ang. *keywords*) języka C#.

Wybierz jedną lub więcej:

- ☐ args
- ☒ string
- ☒ static
- ☐ main
- ☒ void
- ☒ public

Odpowiedź: string, static, void, public

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☐ musi być zadeklarowana w klasie statycznej
- ☐ ma dostęp do niestatycznych składników klasy, w której jest zdefiniowana
- ☐ może być wirtualna
- ☐ żadne z podanych
- ☐ można jej użyć dopiero po utworzeniu obiektu klasy, w której została zdefiniowana
- ☐ może być zdefiniowana w klasie abstrakcyjnej

Odpowiedź: nie, nie, nie, TAK (żdane z podanych), nie, nie

- W ciele metody statycznej, z racji tego iż nie jest wywoływana na rzecz konkretnego obiektu, nie można odwoływać się do składowych niestatycznych. Nie można więc użyć wskaźnika `this`, `self`, `Me` itp.
- Metoda statyczna może wywołać jedynie inne metody statyczne w swojej klasie lub odwoływać się jedynie do [pól statycznych](#) w swojej klasie. Dostęp do pól i metod obiektów przekazywanych jako parametry czy też obiektów i funkcji globalnych następuje tak samo jak w zwykłej funkcji, jednak w przypadku obiektów własnej klasy ma dostęp do składowych prywatnych.
- Metoda statyczna nie może być [metoda wirtualna](#).
- **metod abstrakcyjnych** nie można oznaczać jako **statyczne**

W poniższej definicji klasy konstruktor

W poniższej definicji klasy, który z fragmentów kodu definiuje konstruktor?

```
public class Licznik { // (1)
    int aktualny = 1, krok = 0;

    public void __Construct(int start, int krok) { // (2)
        set(start);
        setKrok(krok);
    }

    public int get() { return aktualny; } // (3)

    public void set(int x) { aktualny = x; } // (4)

    public void setKrok(int s) { krok = s; } // (5)
}
```

Zaznacz właściwe odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Fragment kodu oznaczony jako (1) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (2) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (3) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (4) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (5) jest konstruktorem.
- ☐ W podanej definicji klasy nie ma kodu definiującego konstruktor
- ☐ Konstruktor zostanie automatycznie wygenerowany podczas kompilacji

Odpowiedź: AUTOMATYCZNIE

W poniższej definicji klasy konstruktor

W poniższej definicji klasy, który z fragmentów kodu definiuje konstruktor?

```
public class Licznik { // (1)
    int aktualny, krok;

    public Licznik(int start, int krok) { // (2)
        set(start);
        setKrok(krok);
    }

    public int get() { return aktualny; } // (3)

    public void set(int x) { aktualny = x; } // (4)

    public void setKrok(int s) { krok = s; } // (5)
}
```

Zaznacz właściwe odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Fragment kodu oznaczony jako (1) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (2) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (3) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (4) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (5) jest konstruktorem.
- ☐ W podanej definicji klasy nie ma kodu definiującego konstruktor
- ☐ Konstruktor domyślny zostanie automatycznie wygenerowany podczas kompilacji

Odpowiedź: B->2

Dana jest klasa

Dana jest klasa

```
class K {  
    int i;  
  
    K( int j ) {  
        i = j;  
    }  
  
    void m() {  
        Console.WriteLine(i);  
    }  
  
    // ... tu wstaw kod  
}
```

W miejsce oznaczone komentarzem można wpisać (zaznacz właściwe):

Wybierz jedną lub więcej:

- ☒ `int j = 0;`
- ☐ `int i = 0;`
- ☐ `if (i > 0) Console.WriteLine("i");`
- ☒ `K(){ }`
- ☐ żaden z podanych

Odpowiedź: A D???

W C#.NET nie zostanie przechwycony wyjątek, to co go przechwyci?

W C#.NET, jeśli w programie, w trakcie jego wykonania nie zostanie przechwycony wyjątek, to co go przechwyci?

Wybierz jedną lub więcej:

- ☐ System operacyjny
- ☐ Linker
- ☐ CLR
- ☐ Loader
- ☐ Kompilator

Odpowiedź: Loaderp

Rozważając poniższy kod, która z poniższych instrukcji może być wstawiona

Rozważając poniższy kod:

```
public class ThisUsage {  
    int planets;  
    static int suns;  
  
    public void gaze() {  
        int i;  
        // ... wstaw tu instrukcję  
    }  
}
```

która z poniższych instrukcji może być wstawiona w zaznaczone miejsce?

Wybierz jedną lub więcej:

- ☐ `i = this.planets;`
- ☐ `this = new ThisUsage();`
- ☐ `this.suns = planets;`
- ☐ `i = this.suns;`
- ☐ `this.i = 4;`
- ☐ żadna z podanych

Odpowiedź: A

Przyjmując, że `MojaKlasa` fragment kodu

Przyjmując, że `MojaKlasa` jest niestatyczną klasą, ile obiektów i ile zmiennych referencyjnych zostanie utworzonych w wyniku zadziałania podanego poniżej fragmentu kodu?

```
MojaKlasa x, y;  
x = new MojaKlasa();  
MojaKlasa z = new MojaKlasa();
```

Zaznacz poprawne odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Utworzony zostanie jeden obiekt.
- ☐ Utworzone zostaną dwa obiekty.
- ☐ Utworzone zostaną trzy obiekty.
- ☐ Utworzona zostanie jedna zmienna referencyjna.
- ☐ Utworzone zostaną dwie zmienne referencyjne.
- ☐ Utworzone zostaną trzy zmienne referencyjne.

Odpowiedź: 3 obiekty

Zakładając, że mamy deklarację tablicy zwróci rozmiar

Zakładając, że mamy następującą deklarację tablicy `tab` oraz, że tablica została poprawnie zainicjowana i jej użycie ograniczone jest do przestrzeni nazw `System`, zaznacz wyrażenie, które zwróci rozmiar (liczbę elementów) tej tablicy.

```
using System;  
...  
int[] tab;  
...
```

Wybierz jedną lub więcej:

- ☐ a. `tab.GetLength(0)`
- ☐ b. `tab.GetUpperBound(0)+1`
- ☒ c. `tab.Count()`
- ☐ d. `tab.LongLength`
- ☐ e. żadne z podanych
- ☐ f. `tab.Count`

Odpowiedź: a-> `tab.GetLength(0)` [zwraca długość 1. Wymiar tabeli, Pobiera 32-bitowa liczba całkowita, która reprezentuje liczbę elementów w określonym wymiarze [Array](#).] [**length**.- Pobiera całkowitą liczbę elementów w wszystkie wymiary [Array](#).] [**GetUpperBound** - Pobiera indeks ostatniego elementu określonego wymiaru tablicy.][**GetLongLength** - Pobiera 64-bitowa liczba całkowita, która reprezentuje liczbę elementów w określonym wymiarze [Array](#).]

Wybierz jedną

```
int[] a = new int[] { 5, 6, 7, 8 };
int[] b = new int[] { 1, 2, 3, 4 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

S.Push(Q.Peek()); Q.Dequeue();
S.Push(Q.Peek()); Q.Dequeue();
S.Push(Q.Peek()); Q.Dequeue();
Q.Enqueue(S.Peek()); S.Pop();
Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy `Stack` (stos): `Push` - dodaj (wstaw na stos), `Peek` - odczytaj element wierzchołkowy (do usunięcia), `Pop` - usuń (zdejmij) element wierzchołkowy,
- metody klasy `Queue` (kolejka): `Enqueue` - wstaw do kolejki, `Dequeue` - usuń z kolejki, `Peek` - odczytaj element pierwszy (do usunięcia).

Wybierz jedną odpowiedź:

- ☐ Stos od wierzchołka: 32 | Kolejka: 418765
- ☐ Stos od wierzchołka: 187654 | Kolejka: 32
- ☐ Stos od wierzchołka: 6541 | Kolejka: 3287
- ☐ Stos od wierzchołka: 418765 | Kolejka: 32
- ☐ Stos od wierzchołka: 876541 | Kolejka: 32
- ☐ Stos od wierzchołka: 65 | Kolejka: 873241
- ☐ Stos od wierzchołka: 8765 | Kolejka: 1234
- ☐ Stos od wierzchołka: 6541 | Kolejka: 8732

Odpowiedź: D stos 418765 kolejka 32

Która z podanych deklaracji klasy

Która z podanych deklaracji klasy `Czlowiek` najlepiej opisuje relację "*Pies jest najlepszym przyjacielem człowieka*" (inaczej mówiąc, Człowiek ma Psa, który jest jego najlepszym przyjacielem).

Wybierz jedną odpowiedź:

- ☐ `class Czlowiek : Pies { }`
- ☐ `class Czlowiek { friend Pies najlepszyPrzyjaciol; }`
- ☐ `class Czlowiek { private Pies najlepszyPrzyjaciol; }`
- ☐ `class Czlowiek : friend Pies { }`
- ☐ `class Czlowiek { friend NajlepszyPrzyjaciol pies; }`
- ☐ `class Czlowiek friend Pies { }`
- ☐ `class Czlowiek { private NajlepszyPrzyjaciol pies; }`

Odpowiedź: 3

Dwa fragmenty kodu nazwiemy

Dwa fragmenty kodu nazwiemy równoważnymi, jeżeli w tych samych sytuacjach dają takie same efekty.

Które z podanych poniżej fragmentów kodów są równoważne:

```
//Kod1
if( temperatura > gornyLimit ) {
    if( niebezpieczenstwo ) wywolajAlarm();
} else
    wlaczReaktor();
```

```
//Kod2
if( temperatura > gornyLimit && niebezpieczenstwo )
    wywolajAlarm();
else
    wlaczReaktor();
```

```
//Kod3
if( temperatura <= gornyLimit && !niebezpieczenstwo )
    wlaczReaktor();
else
    wywolajAlarm();
```

Zaznacz zdania prawdziwe

Wybierz jedną odpowiedź:

- ☐ Kod1 i Kod2 są równoważne
- ☐ Kod2 i Kod3 są równoważne
- ☒ Wszystkie kody są sobie równoważne
- ☐ Kod1 i Kod3 są równoważne
- ☐ Żadne kody nie są sobie parami równoważne

Odpowiedź: C

Lista

Dopasuj podstawowy

Dopasuj podstawowy, wbudowany typ danych języka C# do jego opisu

128-bitowa zmiennoprzecinkowa dokładna reprezentacja liczby dziesiętnej ze znakiem, zalecana do wykonywania obliczeń finansowych

decimal ▼

8-bitowa reprezentacja liczby całkowitej bez znaku

byte ▼

64-bitowa reprezentacja liczby całkowitej bez znaku

long ▼

Reprezentacja wartości logicznych `true` oraz `false`

bool ▼

Reprezentacja ciągu znaków o zmiennej długości

string ▼

Typ uniwersalny, będący bazą wszystkich typów - wbudowanych i zdefiniowanych przez użytkownika

var ▼

16-bitowa reprezentacja znaku Unicode

char ▼

Odpowiedź: 1 - decimal, 2 - byte, 3 - ulong, 4 - bool, 5 - StringBuilder, 6 - object, 7 - char

Określ wartość wyrażenia logicznego

Określ wartość wyrażenia logicznego zapisanego w C#. Jeśli jest ono źle skonstruowane, wybierz `błąd kompilacji`.

`new System.Int32(2) == new int(2)`

błąd kompilacji ▼

`' ' == 32`

błąd kompilacji ▼

`true | false`

true ▼

`"1" + " " == "1"`

błąd kompilacji ▼

`0.1 + 0.2 == 0.3`

błąd kompilacji ▼

`(true && false)`

błąd kompilacji ▼

Odpowiedź: 1 - błąd

2 - true

3 - true

4 - błąd false

5 - false

6 - false

Określ wartość wyrażenia logicznego

Określ wartość wyrażenia logicznego zapisanego w C#. Jeśli jest ono źle skonstruowane, wybierz **błąd kompilacji**.

`2f + 2d == 4`

błąd kompilacji ▼

`(false == false)`

true ▼

`(new int()) is ValueType`

true ▼

`"1" + '' == "1"`

błąd kompilacji ▼

`new int() is null`

błąd kompilacji ▼

`(new int()) == 0`

błąd kompilacji ▼

Odpowiedź:

- 1 - true
- 2 - true
- 3- true
- 4 -false
- 5 - błąd
- 6- true

Przeciągnij

Liczba pierwsza to taka

Liczba pierwsza to taka liczba naturalna większa od 1, która dzieli się tylko przez 1 i samą siebie. Oto kilka początkowych liczb pierwszych: 2, 3, 5, 7, 11, 13, 17, 19, ...

Aby określić, czy dana liczba jest pierwsza należy zbadać jej dzielniki. Dla zadanej liczby n sprawdzamy kolejne liczby naturalne mniejsze od niej. Jeśli któraś z tych liczb jest dzielnikiem n , oznacza to, że n nie jest liczbą pierwszą.

Algorytm ten można zoptymalizować - wystarczy sprawdzać liczby z przedziału $[2, \sqrt{n}]$.

Napisz w C# funkcję o nazwie `JestPierwsza`, która dla zadanej wartości `n` typu `int` zwróci prawdę, gdy `n` jest liczbą pierwszą oraz fałsz w przeciwnym przypadku.

Wykorzystaj podany powyżej pomysł algorytmu zoptymalizowanego oraz podany poniżej szkielet funkcji, przeciągając i upuszczając odpowiednie bloki kodu w odpowiednie miejsca.

Dane bloki kodu mogą być wykorzystane raz, wiele razy lub ani razu. Wszystkie puste pola w kodzie algorytmu muszą mieć przypisane właściwe bloki kodu.

```
public static  JestPierwsza( int n )
{
    if( n<=1 ) throw new ArgumentOutOfRangeException();
     ;
    while(  )
    {
        if(  ) return  ;
         ;
    }
    return  ;
}
```

true	$n \% i > 0$	$n \% i == 0$	bool	$n == i*i$	int i = 1
false	i++	$i*i <= n$	$i \% n > 0$	int	$n / i == 0$
$i \% n == 0$	$i >= 2 \ \&\& \ i < \text{Math.Sqrt}(n)$	int i = 2			

Odpowiedź:

1) bool	
2) int i = 2	
3) i * i <= n	
4) n % i == 0	5) false
6) i++	
7) true	

Jaki będzie stan końcowy stosu

Jaki będzie stan końcowy stosu (zmienna **S**) oraz kolejki (zmienna **Q**) po wykonaniu następującego fragmentu programu napisanego w C#:

```
int[] a = new int[] { 1, 2, 3, 4 };
int[] b = new int[] { 5, 6, 7, 8 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek());
S.Push(Q.Peek()); Q.Dequeue();
Q.Enqueue(S.Peek()); S.Pop();
Q.Enqueue(S.Peek());
S.Push(Q.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy **Stack** (stos): **Push** - dodaj (wstaw na stos), **Peek** - odczytaj element wierzchołkowy (do usunięcia), **Pop** - usuń (zdejmij) element wierzchołkowy,
- metody klasy **Queue** (kolejka): **Enqueue** - wstaw do kolejki, **Dequeue** - usuń z kolejki, **Peek** - odczytaj element pierwszy (do usunięcia).

Przeciągnij właściwe markery w odpowiednie miejsca. W pustych, nie wypełnionych komórkach, umieść marker z kreską [-].

Jeśli uruchomienie kodu spowoduje pojawienie się wyjątku (stos pusty, kolejka pusta) - we wszystkich komórkach kolejki i stosu umieść marker z kreską [-].

- Stos od wierzchołka:

--	--	--	--	--	--	--	--
- Kolejka od początku:

--	--	--	--	--	--	--	--

1	2	3	4	5	6	7	8	-
---	---	---	---	---	---	---	---	---

Odpowiedzi: stos: 65321???, kolejka:78455 ???

Ile razy wykona się

Ile razy wykona się instrukcja zawarta w podanej pętli?
Program zapisany jest w pseudokodzie.

```
x ← 1
do
  instrukcja
  x ← x + 5
while (x ≤ 5)
```

Podaj liczbę wywołań instrukcji

```
int x = 1;
do
{
    x = x + 5;
    Console.WriteLine("o");
} while (x <= 5);

Console.ReadKey();
```

Które z twierdzeń jest prawdziwe:



Odp: Index out of bounds Remaining program

```
using System;
using System.Collections.Generic;
using System.Collections;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp3
{
    class Program
    {
        static void Main(string[] args)
        {
            int k = 1;
            int i = 2;
            while (i < 10)
            {
                if (i % 2 == 0)
                {
                    k = k++;
                    Console.WriteLine("linia");
                }
            }
        }
    }
}
```

```

        i++;
    }

    int x = 1;
    do
    {
        x = x + 4;
        Console.WriteLine("pseudokod");
    } while (x <= 7);

    int fun(int n)
    {
        if (n < 2) return n;
        if (n % 2 == 0) return fun(n - 3) + 1;
        else return fun(n + 1) + 1;
    }
    Console.WriteLine(fun(5));
    Console.WriteLine(fun(6));
    Console.WriteLine(fun(7));
    Console.WriteLine("mm");

    int fun2(int m)
    {
        if (m < 2) return m;
        else return fun2(m - 1);
    }
    Console.WriteLine(fun2(6));
    Console.WriteLine(fun2(7));
    Console.WriteLine(fun2(8));

    Console.WriteLine();
    Console.WriteLine();
    Console.WriteLine("wth");

    int[] g = new int[] {5,6,7,8 };
    int[] f = new int[] {1,2,3, 4 };

    Stack<int> S= new Stack<int>(g);
    Queue<int> Q= new Queue<int>(f);

    S.Push(Q.Peek()); Q.Dequeue();

    /*
    for(int ei=0; ei<g.Length; ei++)
    {
        Console.WriteLine(g[ei]);
    }*/
    Console.WriteLine("Stos");

    foreach(int ed in S)
    {
        Console.WriteLine(ed);
    }
    Console.WriteLine();
    Console.WriteLine("Kolejka");
    foreach (int ex in Q)
    {

```

```

        Console.Write(ex);
    }
    Console.WriteLine();
    Console.WriteLine();
    S.Push(Q.Peek()); Q.Dequeue();
    Console.WriteLine("Stos");

    foreach (int ed in S)
    {
        Console.Write(ed);
    }
    Console.WriteLine();
    Console.WriteLine("Kolejka");
    foreach (int ex in Q)
    {
        Console.Write(ex);
    }
    Console.WriteLine();
    Console.WriteLine();
    S.Push(Q.Peek()); Q.Dequeue();
    Console.WriteLine("Stos");

    foreach (int ed in S)
    {
        Console.Write(ed);
    }
    Console.WriteLine();
    Console.WriteLine("Kolejka");
    foreach (int ex in Q)
    {
        Console.Write(ex);
    }
    Console.WriteLine();
    Console.WriteLine();
    Q.Enqueue(S.Peek()); S.Pop();
    Console.WriteLine("Stos");

    foreach (int ed in S)
    {
        Console.Write(ed);
    }
    Console.WriteLine();
    Console.WriteLine("Kolejka");
    foreach (int ex in Q)
    {
        Console.Write(ex);
    }

```

```

Console.WriteLine();
Console.WriteLine();
S.Push(Q.Peek()); Q.Dequeue();
Console.WriteLine("Stos");

foreach (int ed in S)
{
    Console.Write(ed);
}
Console.WriteLine();
Console.WriteLine("Kolejka");
foreach (int ex in Q)
{
    Console.Write(ex);
}
Console.WriteLine();
Console.WriteLine();
MojaKlasa o;
MojaKlasa p;
o = new MojaKlasa();
MojaKlasa r = new MojaKlasa();
Console.WriteLine("df");
/* bool coTo;

    if ( ' ' == 32)
    {
        coTo = true;
        Console.WriteLine(coTo);
    }
    else
        Console.WriteLine(coTo);*/

/*bool coTo = false;

if (coTo == true | coTo == false)
{
    coTo = true;
    Console.WriteLine(coTo);
}
else
    Console.WriteLine(coTo);
*/
/*bool coTo = false;

if (0.1 + 0.2 == 0.3)
{
    coTo = true;
    Console.WriteLine(coTo);
}
else
    Console.WriteLine(coTo);*/
/*
        bool coTo = false;

        if (coTo == true && coTo == false)
        {
            coTo = true;
            Console.WriteLine(coTo);
        }
        else
            Console.WriteLine(coTo);
        */

```

```

        bool coTo = false;

        if ("1" + ' ' == "1")
        {
            coTo = true;
            Console.WriteLine(coTo);
        }
        else
            Console.WriteLine(coTo);

        Console.WriteLine();
        Console.WriteLine();
        Console.WriteLine();

        int[] g2 = new int[] { 1, 2, 3, 4 };
        int[] f2 = new int[] { 5, 6, 7, 8 };

        Stack<int> S2 = new Stack<int>(g2);
        Queue<int> Q2 = new Queue<int>(f2);

        Q2.Enqueue(S2.Peek()); S2.Pop();
        S2.Push(Q2.Peek());
        S2.Push(Q2.Peek()); Q2.Dequeue();
        Q2.Enqueue(S2.Peek()); S2.Pop();
        Q2.Enqueue(S2.Peek());
        S2.Push(Q2.Peek()); Q2.Dequeue();

        /*
        for(int ei=0; ei<g.Length; ei++)
        {
            Console.WriteLine(g[ei]);
        }
        */
        Console.WriteLine("Stos2");

        foreach (int ed in S2)
        {
            Console.Write(ed);
        }
        Console.WriteLine();
        Console.WriteLine("Kolejka2");
        foreach (int ex in Q2)
        {
            Console.Write(ex);
        }
        Console.WriteLine();
        Console.WriteLine();
        Console.WriteLine();
        Console.WriteLine();

        int iud = -1;
        int ksy = 1 - iud;
        Console.WriteLine(ksy);
        Console.ReadKey();
    }
}

public class K
{
    int i;
    K (int j)

```



```

        {
            i = j;
        }

        void m()
        {
            Console.WriteLine(i);
        }

        K() { }
    }

    public class ThisUsage
    {
        int planets;
        static int suns;

        public void gaze()
        {
            int i;
            i = this.planets;
        }
    }

    public class MojaKlasa
    {
    }
}

```

Za podanie odpowiedzi brawa dla:

- Morasiu
- Dawida Burnat
- Macieja Romanowskiego
- [tu możesz być ty!]
- Katarzyny Twardowskiej (MrocznyNietoperzBatman124)

Zapraszam do dodawania pytań na Discord i odpowiedzi w niniejszym arkuszu~!

<https://e.wsei.edu.pl/mod/quiz/view.php?id=5604>

cs201-stac

Do wpisania

Jaka będzie wartość zmiennej

Jaka będzie wartość zmiennej `k` po wykonaniu podanego fragmentu kodu

```
01. int k = 1;
02. int i = 2;
03. while( i < 10 )
04. {
05.     if( i % 2 == 0 )
06.         k = k++;
07.     i++;
08. }
```

Wpisz wartość.

Odpowiedź:

Odpowiedź: 1

Jaka będzie wartość zmiennej

`n` po wykonaniu poniższego kodu, dla `i = 2` oraz `n = 3`.

Zakładamy, że wszystkie zmienne są poprawnie zadeklarowane

```
while (!(i > 4))
{
    n += i;
    i++;
}
```

Wpisz wartość liczbową. Jeśli program się zapętlą, wpisz `-999999` (nie używaj zbędnych symboli i spacji - poprawia automat).

Jaka będzie wartość zmiennej

`n` po wykonaniu poniższego kodu.

Zakładamy, że wszystkie zmienne są poprawnie zadeklarowane

```
int n = 2;
for (int i = 1; i <= 3; i++)
{
    for (int j = 1; j <= 4; j++)
    {
        if (i != j)
        {
            n--;
        }
    }
}
```

Wpisz wartość liczbową. Jeśli program się zapętlą, wpisz `-999999` (nie używaj zbędnych symboli i spacji - poprawia automat).

Odp: -7

Ile razy wykona się

Ile razy wykona się instrukcja zawarta w podanej pętli?

Program zapisany jest w pseudokodzie.

```
x ← 1  
do  
  instrukcja  
  x ← x + 4  
while (x ≤ 7)
```

Podaj liczbę wywołań instrukcji

Odpowiedź: 2

Ile razy wykona się

Ile razy wykona się instrukcja zawarta w podanej pętli?

Program zapisany jest w pseudokodzie.

```
x ← 2  
do  
  instrukcja  
  x ← x + 2  
while (x < 5)
```

Podaj liczbę wywołań instrukcji

Odpowiedź: 2

Ile wykona się

```
int x = 1;  
do  
{  
  x = x + 5;  
  Console.WriteLine("o");  
} while (x <= 5);
```

Ile razy wykonana się instrukcja zawarta w podanej pętli?
Program zapisany jest w pseudokodzie.

```
x ← 1
do
  instrukcja
  x ← x + 5
while (x ≤ 5)
```

Podaj liczbę wywołań instrukcji

Odp: 1

Ustal, co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 5, 6, 7 podanych kolejno:

```
int fun(int n) {
    if (n < 2) return n;
    if (n % 2 == 0) return fun(n-3) + 1;
    else return fun(n + 1)+1;
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).

Jeśli funkcja się zapętlą w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź:

Odpowiedź: 5;4;7

Ustal, co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 6, 7, 8 podanych kolejno:

```
int fun(int n) {  
    if (n < 2) return n;  
    else return fun(n - 1);  
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).
Jeśli funkcja się zapęła w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź: 1,1,1,

Ustal, co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 7, 8, 9 podanych kolejno:

```
int fun(int n){  
    if (n < 2) return n;  
    if (n % 2 == 0) return fun(n - 1) + 1;  
    else return fun(n / 2);  
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).

Jeśli funkcja się zapęła w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź: 1;2;2

Odpowiedź: 1;2;2

Ustal co zwraca poniższa

Ustal, co zwraca poniższa funkcja rekurencyjna dla wartości 7, 8, 9 podanych kolejno:

```
int fun(int n) {  
    if (n < 2) return n;  
    if (n % 2 == 1) return fun(n - 1) + 1;  
    else return fun(n / 2);  
}
```

Wpisz do pola tekstowego zwracane wartości kolejno, oddzielając je średnikami (bez zbędnych spacji - ocenia automat).

Jeśli funkcja się zapęła w którymś przypadku, zapisz odpowiednio symbol: **N/A**

Przykłady poprawnie uformowanych odpowiedzi:

- 2;3;4

albo

- 3;N/A;4

Odpowiedź: 3;1;2

Podaj w systemie

Podaj, w systemie dziesiętkowym, wartość liczby zakodowanej w systemie **uzupełnienie-do-dwa**:

$$(00001110)_{u_2} = (\dots)_{10}$$

Odpowiedź:

Odpowiedź: 14

Podaj wyraz zakodowany w UTF-8

Podaj wyraz zakodowany w UTF-8 (bez BOM). Wielkość liter jest istotna .

Odpowiedź:

Odpowiedź: wróżka(konwertując do ASCII w notepadzie++)

Podaj wyraz zakodowany w UTF-8 (bez BOM). Wielkość liter jest istotna .

Odpowiedź:

odp: wędka

Jeśli jesteś nowym klientem banku

Jeśli jesteś nowym klientem otwierającym konto z kartą kredytową, otrzymujesz 15% zniżkę na dzisiejsze zakupy. Jeśli jesteś stałym klientem banku i masz kartę lojalnościową, otrzymujesz 10% zniżkę. Jeśli masz kupon polecający, otrzymujesz 20% zniżki na dzisiejsze zakupy, ale kupon nie może być wykorzystany ze zniżką dla nowego klienta. Zniżki się sumują (jeśli to nie jest wykluczone powyższym regulaminem).

Uzupełnij tablicę decyzyjną (przeciągnij i upuść markery na odpowiednie pola):

Warunki	Reguła 1	Reguła 2	Reguła 3	Reguła 4	Reguła 5	Reguła 6
Nowy klient (15%)	Tak	<input type="text" value="3"/>	<input type="text" value="6"/>	<input type="text" value="9"/>	Nie	<input type="text" value="14"/>
Karta lojalnościowa (10%)	<input type="text" value="1"/>	<input type="text" value="4"/>	<input type="text" value="7"/>	<input type="text" value="10"/>	<input type="text" value="12"/>	<input type="text" value="15"/>
Kupon (20%)	<input type="text" value="2"/>	<input type="text" value="5"/>	<input type="text" value="8"/>	<input type="text" value="11"/>	<input type="text" value="13"/>	<input type="text" value="16"/>
Działanie						
Zniżka (%)	20	15	30	10	20	0

Odpowiedź:

Jeśli jesteś nowym klientem otwierającym konto z kartą kredytową, otrzymujesz 15% zniżkę na dzisiejsze zakupy. Jeśli jesteś stałym klientem banku i masz kartę lojalnościową, otrzymujesz 10% zniżkę. Jeśli masz kupon polecający, otrzymujesz 20% zniżki na dzisiejsze zakupy, ale kupon nie może być wykorzystany ze zniżką dla nowego klienta. Zniżki się sumują (jeśli to nie jest wykluczone powyższym regulaminem).

Uzupełnij tablicę decyzyjną (przeciągnij i upuść markery na odpowiednie pola):

Warunki	Reguła 1	Reguła 2	Reguła 3	Reguła 4	Reguła 5	Reguła 6
Nowy klient (15%)	Tak	<div>Tak</div>	<div>Nie</div>	<div>Nie</div>	Nie	<div>Nie</div>
Karta lojalnościowa (10%)	<div>Nie</div>	<div>Nie</div>	<div>Tak</div>	<div>Tak</div>	<div>Nie</div>	<div>Nie</div>
Kupon (20%)	<div>Tak</div>	<div>Nie</div>	<div>Tak</div>	<div>Nie</div>	<div>Tak</div>	<div>Nie</div>
Działanie						
Zniżka (%)	20	15	30	10	20	0

Tak

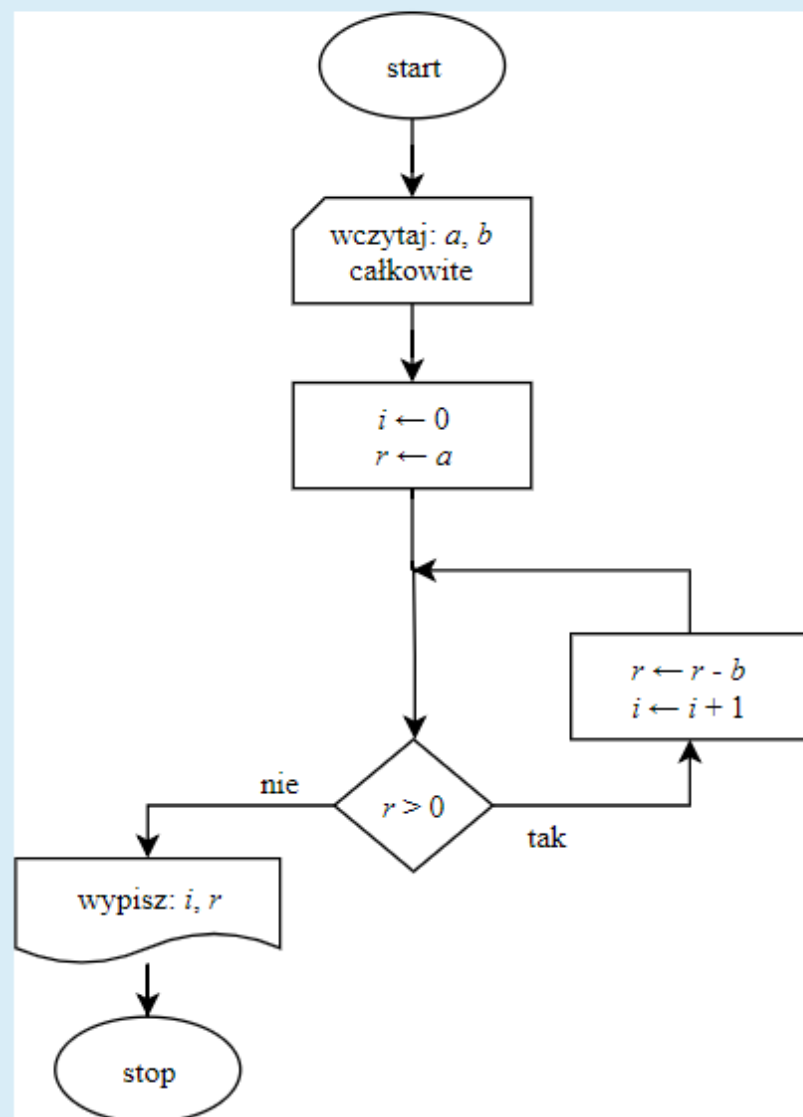
Nie

--

Reguła 1	Reguła 2	Reguła 3	Reguła 4	Reguła 5	Reguła 6
Tak	Tak	Nie	Nie	Nie	Nie
Nie	Nie	Tak	Tak	Nie	Nie
Tak	Nie	Tak	Nie	Tak	Nie

Algorytm jaka będzie wartość zmiennej i

Jaka będzie wartość zmiennej i po zakończeniu działania algorytmu, jeśli na wejściu wczytano wartości całkowite $a = 23$ oraz $b = -8$.

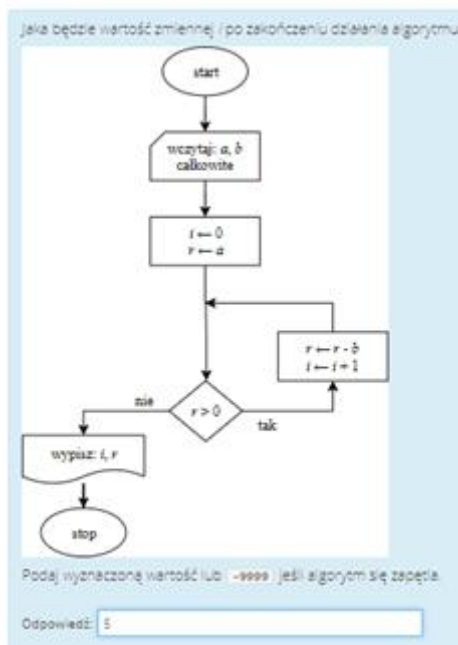


Podaj wyznaczoną wartość lub jeśli algorytm się zapętla.

Odpowiedź:

Odpowiedź: -9999

Algorytm jaka będzie wartość zmiennej i



Odp: 5

Zaznacz kilka

Zaznacz słowa zarezerwowane

Zaznacz słowa zarezerwowane (ang. *keywords*) języka C#.

Wybierz jedną lub więcej:

- ☐ args
- ☒ string
- ☒ static
- ☐ main
- ☒ void
- ☒ public

Odpowiedź: string, static, void, public

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☐ musi być zadeklarowana w klasie statycznej
- ☐ ma dostęp do niestatycznych składników klasy, w której jest zdefiniowana
- ☐ może być wirtualna
- ☐ żadne z podanych
- ☐ można jej użyć dopiero po utworzeniu obiektu klasy, w której została zdefiniowana
- ☐ może być zdefiniowana w klasie abstrakcyjnej

Odpowiedź: D

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☐ może być przeciążana
- ☒ nie może być składnikiem interfejsu
- ☒ może być chroniona (`protected`)
- ☒ nie ma dostępu do niestatycznych składników klasy, w której jest zdefiniowana
- ☐ żadne z podanych
- ☒ można jej użyć nawet wtedy, gdy nie ma utworzonego obiektu klasy, w której została zdefiniowana

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☒ żadne z podanych
- ☐ nie może być składnikiem typu wartościowego
- ☐ musi być zadeklarowana w klasie statycznej
- ☐ może być przesłaniana
- ☐ może być abstrakcyjna
- ☐ można jej użyć dopiero po utworzeniu obiektu klasy, w której została zdefiniowana

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☐ żadne z podanych
- ☒ może być przesłaniana
- ☐ można jej użyć dopiero po utworzeniu obiektu klasy, w której została zdefiniowana
- ☒ nie może być składnikiem interfejsu
- ☐ nie może być prywatna
- ☒ można jej użyć nawet wtedy, gdy nie ma utworzonego obiektu klasy, w której została zdefiniowana

Zaznacz poprawne stwierdzenia metody statycznej

Zaznacz poprawne stwierdzenia, w odniesieniu do metody statycznej

Wybierz jedną lub więcej:

- ☐ żadne z podanych
- ☒ nie ma dostępu do instancji dowolnego obiektu, o ile nie jest on jawnie przekazany jako jej parametr
- ☐ może być przeciążana
- ☐ można jej użyć dopiero po utworzeniu obiektu klasy, w której została zdefiniowana
- ☐ musi być zadeklarowana w klasie statycznej
- ☒ może być składnikiem typu wartościowego

W poniższej definicji klasy konstruktor

W poniższej definicji klasy, który z fragmentów kodu definiuje konstruktor?

```
public class Licznik {                                // (1)
    int aktualny = 1, krok = 0;

    public void __Construct(int start, int krok) {    // (2)
        set(start);
        setKrok(krok);
    }

    public int get() { return aktualny; }             // (3)

    public void set(int x) { aktualny = x; }          // (4)

    public void setKrok(int s) { krok = s; }          // (5)
}
```

Zaznacz właściwe odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Fragment kodu oznaczony jako (1) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (2) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (3) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (4) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (5) jest konstruktorem.
- ☐ W podanej definicji klasy nie ma kodu definiującego konstruktor
- ☐ Konstruktor zostanie automatycznie wygenerowany podczas kompilacji

Odpowiedź: F G???

Które z podanych stwierdzeń jest prawdziwe

Które z podanych stwierdzeń jest prawdziwe dla poniższego programu?

```
using System;
namespace ConsoleApplication
{
    class MyProgram
    {
        static void Main(string[] args)
        {
            int index = 6;
            int val = 44;
            int[] a = new int[6];
            try
            {
                a[index] = val;
            }
            catch (IndexOutOfRangeException e)
            {
                Console.Write("Index out of bounds ");
            }
            Console.Write("Remaining program");
        }
    }
}
```

Wybierz jedną lub więcej:

- ☒ Wypisane zostanie: *Index out of bounds Remaining program*
- ☐ Wypisane zostanie: *Index out of bounds*
- ☐ Błąd kompilacji / błąd syntaktyczny
- ☐ Nic nie zostanie wypisane
- ☐ Wypisane zostanie: *Remaining program*
- ☐ Wartość `44` zostanie przypisana do `a[6]`.

Odpowiedź: Wypisane zostanie: *Index out of bounds Remaining program*

W poniższej definicji klasy konstruktor

W poniższej definicji klasy, który z fragmentów kodu definiuje konstruktor?

```
public class Licznik { // (1)
    int aktualny, krok;

    public Licznik(int start, int krok) { // (2)
        set(start);
        setKrok(krok);
    }

    public int get() { return aktualny; } // (3)

    public void set(int x) { aktualny = x; } // (4)

    public void setKrok(int s) { krok = s; } // (5)
}
```

Zaznacz właściwe odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Fragment kodu oznaczony jako (1) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (2) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (3) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (4) jest konstruktorem.
- ☐ Fragment kodu oznaczony jako (5) jest konstruktorem.
- ☐ W podanej definicji klasy nie ma kodu definiującego konstruktor
- ☐ Konstruktor domyślny zostanie automatycznie wygenerowany podczas kompilacji

Odpowiedź: B

Dana jest klasa

Dana jest klasa

```
class K {  
    int i;  
  
    K( int j ) {  
        i = j;  
    }  
  
    void m() {  
        Console.WriteLine(i);  
    }  
  
    // ... tu wstaw kod  
}
```

W miejsce oznaczone komentarzem można wpisać (zaznacz właściwe):

Wybierz jedną lub więcej:

- ☒ `int j = 0;`
- ☐ `int i = 0;`
- ☐ `if (i > 0) Console.WriteLine("i");`
- ☒ `K(){ }`
- ☐ żaden z podanych

Odpowiedź: A D???

W C#.NET nie zostanie przechwycony wyjątek, to co go przechwyci?

W C#.NET, jeśli w programie, w trakcie jego wykonania nie zostanie przechwycony wyjątek, to co go przechwyci?

Wybierz jedną lub więcej:

- ☐ System operacyjny
- ☐ Linker
- ☐ CLR
- ☐ Loader
- ☐ Kompilator

Odpowiedź: ??? Loader?

Rozważając poniższy kod, która z poniższych instrukcji może być wstawiona

Rozważając poniższy kod:

```
public class ThisUsage {  
    int planets;  
    static int suns;  
  
    public void gaze() {  
        int i;  
        // ... wstaw tu instrukcję  
    }  
}
```

która z poniższych instrukcji może być wstawiona w zaznaczone miejsce?

Wybierz jedną lub więcej:

- ☐ `i = this.planets;`
- ☐ `this = new ThisUsage();`
- ☐ `this.suns = planets;`
- ☐ `i = this.suns;`
- ☐ `this.i = 4;`
- ☐ żadna z podanych

Odpowiedź: A ???

Przyjmując, że MojaKlasa fragment kodu

Przyjmując, że `MojaKlasa` jest niestaticzną klasą, ile obiektów i ile zmiennych referencyjnych zostanie utworzonych w wyniku zadziałania podanego poniżej fragmentu kodu?

```
MojaKlasa x, y;  
x = new MojaKlasa();  
MojaKlasa z = new MojaKlasa();
```

Zaznacz poprawne odpowiedzi.

Wybierz jedną lub więcej:

- ☐ Utworzony zostanie jeden obiekt.
- ☐ Utworzone zostaną dwa obiekty.
- ☐ Utworzone zostaną trzy obiekty.
- ☐ Utworzona zostanie jedna zmienna referencyjna.
- ☐ Utworzone zostaną dwie zmienne referencyjne.
- ☐ Utworzone zostaną trzy zmienne referencyjne.

Odpowiedź: ???

Zakładając, że mamy deklarację tablicy zwróci rozmiar

Zakładając, że mamy następującą deklarację tablicy `tab` oraz, że tablica została poprawnie zainicjowana i jej użycie ograniczone jest do przestrzeni nazw `System`, zaznacz wyrażenie, które zwróci rozmiar (liczbę elementów) tej tablicy.

```
using System;
...
int[] tab;
...
```

Wybierz jedną lub więcej:

- ☐ a. `tab.GetLength(0)`
- ☐ b. `tab.GetUpperBound(0)+1`
- ☒ c. `tab.Count()`
- ☐ d. `tab.LongLength`
- ☐ e. żadne z podanych
- ☐ f. `tab.Count`

Odpowiedź: A B C D

Zakładając, że mamy deklarację tablicy zwróci rozmiar

Zakładając, że mamy następującą deklarację tablicy `tab` oraz, że tablica została poprawnie zainicjowana i jej użycie ograniczone jest do przestrzeni nazw `System`, zaznacz wyrażenie, które zwróci rozmiar (liczbę elementów) tej tablicy.

```
using System;
...
int[] tab;
...
```

Wybierz jedną lub więcej:

- ☐ a. `tab.length`
- ☐ b. żadne z podanych
- ☒ c. `tab.Length`
- ☐ d. `tab.Size`
- ☐ e. `tab[].Size()`
- ☒ f. `tab.GetLength(0)`

Które z podanych stwierdzeń jest prawdziwe

Które z podanych stwierdzeń jest prawdziwe w odniesieniu do wyjątków w C#.NET?

Wybierz jedną lub więcej:

- ☐ Pojawiają się w trakcie linkowania
- ☐ Pojawiają się w trakcie wykonania programu
- ☐ Pojawiają się w trakcie ładowania programu
- ☐ Pojawiają się podczas kompilacji kodu
- ☐ Pojawiają się w trakcie kompilacji *Just-In-Time*

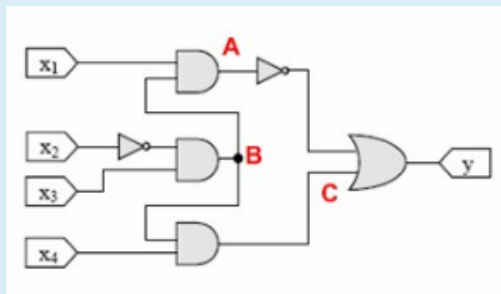
Odpowiedź: ???

Pytanie odnosi się do podanego schematu bramek logicznych

Pytanie odnosi się do podanego schematu bramek logicznych.

Jaki będzie stan w punktach A, B, C oznaczonych na rysunku kolorem czerwonym, jeśli:

- $x_1 = 0$
- $x_2 = 1$
- $x_3 = 0$
- $x_4 = 1$



Wybierz jedną odpowiedź:

- ☐ A=0; B=1; C=1
- ☐ A=0; B=0; C=0
- ☐ żadna z podanych odpowiedzi
- ☐ A=1; B=1; C=0
- ☐ A=1; B=0; C=0
- ☐ A=0; B=0; C=1
- ☐ A=1; B=0; C=1
- ☐ A=0; B=1; C=0

Odpowiedź: ???

Rozważ następujący kod

Rozważ następujący kod:

```
public class Przyklad
{
    public static void Main(string[] args)
    {
        int n;
        Console.WriteLine("n=" + n);
    }
}
```

Jaki będzie wynik po wyprodukowaniu kodu (kompilacja, uruchomienie)?

Wybierz jedną lub więcej:

- ☐ Kompilator wykaże błąd.
- ☐ Kod zostanie skompilowany, ale nic nie zostanie wyświetlone.
- ☐ Wyświetlony zostanie napis `n=`
- ☐ Wyświetlony zostanie napis `n=n`
- ☐ Wyświetlony zostanie napis `n=0`

Odpowiedź: a ???

Dane są klasy K oraz L

Dane są klasy: `K` oraz `L`. Chcemy, aby projektowana przez nas klasa `X` miała własności obu tych klas.

Który z wariantów deklaracji klasy `X` jest poprawny?

Wybierz jedną lub więcej:

- ☐ `class X : K implements L { }`
- ☐ `class X : K : L { }`
- ☐ `class X : K, L { }`
- ☐ `class X : K, : L { }`
- ☐ Żadne z podanych: dziedziczenie wielokrotne nie jest w C# dozwolone.

Odpowiedź: ???

Wybierz jedną

```
int[] a = new int[] { 5, 6, 7, 8 };
int[] b = new int[] { 1, 2, 3, 4 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

S.Push(Q.Peek()); Q.Dequeue();
S.Push(Q.Peek()); Q.Dequeue();
S.Push(Q.Peek()); Q.Dequeue();
Q.Enqueue(S.Peek()); S.Pop();
Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy `Stack` (stos): `Push` - dodaj (wstaw na stos), `Peek` - odczytaj element wierzchołkowy (do usunięcia), `Pop` - usuń (zdejmij) element wierzchołkowy,
- metody klasy `Queue` (kolejka): `Enqueue` - wstaw do kolejki, `Dequeue` - usuń z kolejki, `Peek` - odczytaj element pierwszy (do usunięcia).

Wybierz jedną odpowiedź:

- ☐ Stos od wierzchołka: 32 | Kolejka: 418765
- ☐ Stos od wierzchołka: 187654 | Kolejka: 32
- ☐ Stos od wierzchołka: 6541 | Kolejka: 3287
- ☐ Stos od wierzchołka: 418765 | Kolejka: 32
- ☐ Stos od wierzchołka: 876541 | Kolejka: 32
- ☐ Stos od wierzchołka: 65 | Kolejka: 873241
- ☐ Stos od wierzchołka: 8765 | Kolejka: 1234
- ☐ Stos od wierzchołka: 6541 | Kolejka: 8732

Odpowiedź: D ???

Która z podanych deklaracji klasy

która z podanych deklaracji klasy `Czlowiek` najlepiej opisuje relację "*Pies jest najlepszym przyjacielem człowieka*" (inaczej mówiąc, Człowiek ma Psa, który jest jego najlepszym przyjacielem).

Wybierz jedną odpowiedź:

- ☐ `class Czlowiek : Pies { }`
- ☐ `class Czlowiek { friend Pies najlepszyPrzyjaciol; }`
- ☐ `class Czlowiek { private Pies najlepszyPrzyjaciol; }`
- ☐ `class Czlowiek : friend Pies { }`
- ☐ `class Czlowiek { friend NajlepszyPrzyjaciol pies; }`
- ☐ `class Czlowiek friend Pies { }`
- ☐ `class Czlowiek { private NajlepszyPrzyjaciol pies; }`

Odpowiedź: ???

Dwa fragmenty kodu nazwiemy

Dwa fragmenty kodu nazwiemy równoważnymi, jeżeli w tych samych sytuacjach dają takie same efekty.

Które z podanych poniżej fragmentów kodów są równoważne:

```
//Kod1
if( temperatura > gornyLimit ) {
    if( niebezpieczenstwo ) wywolajAlarm();
} else
    wlaczReaktor();
```

```
//Kod2
if( temperatura > gornyLimit && niebezpieczenstwo )
    wywolajAlarm();
else
    wlaczReaktor();
```

```
//Kod3
if( temperatura <= gornyLimit && !niebezpieczenstwo )
    wlaczReaktor();
else
    wywolajAlarm();
```

Zaznacz zdania prawdziwe

Wybierz jedną odpowiedź:

- ☐ Kod1 i Kod2 są równoważne
- ☐ Kod2 i Kod3 są równoważne
- ☒ Wszystkie kody są sobie równoważne
- ☐ Kod1 i Kod3 są równoważne
- ☐ Żadne kody nie są sobie parami równoważne

Odpowiedź: C ???

Lista

Dopasuj podstawowy

Dopasuj podstawowy, wbudowany typ danych języka C# do jego opisu

128-bitowa zmiennoprzecinkowa dokładna reprezentacja liczby dziesiętnej ze znakiem, zalecana do wykonywania obliczeń finansowych

decimal ▼

8-bitowa reprezentacja liczby całkowitej bez znaku

byte ▼

64-bitowa reprezentacja liczby całkowitej bez znaku

long ▼

Reprezentacja wartości logicznych `true` oraz `false`

bool ▼

Reprezentacja ciągu znaków o zmiennej długości

string ▼

Typ uniwersalny, będący bazą wszystkich typów - wbudowanych i zdefiniowanych przez użytkownika

var ▼

16-bitowa reprezentacja znaku Unicode

char ▼

Odpowiedź: 1 - decimal, 2 - byte, 3 - ulong, 4 - bool, 5 - StringBuilder, 6 - object, 7 - char

Dopasuj podstawowy

Dopasuj podstawowy, wbudowany typ danych języka C# do jego opisu

Typ uniwersalny, będący bazą wszystkich typów - wbudowanych i zdefiniowanych przez użytkownika

var ▼

128-bitowa zmiennoprzecinkowa dokładna reprezentacja liczby dziesiętnej ze znakiem, zalecana do wykonywania obliczeń finansowych

decimal ▼

Reprezentacja wartości logicznych `true` oraz `false`

bool ▼

64-bitowa zmiennoprzecinkowa reprezentacja liczby zgodna ze standardem IEEE 754

double ▼

32-bitowa zmiennoprzecinkowa reprezentacja liczby zgodna ze standardem IEEE 754

float ▼

Reprezentacja ciągu znaków o zmiennej długości

string ▼

16-bitowa reprezentacja znaku Unicode

char ▼

Odpowiedź: 1 - var, 2 - decimal, 3 - bool, 4 - double, 5 - float, 6 - string, 7 - char

Dopasuj podstawowy, wbudowany typ danych języka C# do jego opisu

32-bitowa reprezentacja liczby całkowitej ze znakiem	int ▼
64-bitowa reprezentacja liczby całkowitej ze znakiem	long ▼
16-bitowa reprezentacja liczby całkowitej ze znakiem	short ▼
16-bitowa reprezentacja liczby całkowitej bez znaku	ushort ▼
8-bitowa reprezentacja liczby całkowitej ze znakiem	sbyte ▼
32-bitowa reprezentacja liczby całkowitej bez znaku	uint ▼
8-bitowa reprezentacja liczby całkowitej bez znaku	byte ▼

1-int; 2-long; 3- short; 4-ushort; 5-sbyte; 6-uint; 7-byte

Określ wartość wyrażenia logicznego

Określ wartość wyrażenia logicznego zapisanego w C#. Jeśli jest ono źle skonstruowane, wybierz `błąd kompilacji`.

<code>new System.Int32(2) == new int(2)</code>	<input type="text" value="błąd kompilacji"/>
<code>' ' == 32</code>	<input type="text" value="błąd kompilacji"/>
<code>true false</code>	<input type="text" value="true"/>
<code>"1" + " " == "1"</code>	<input type="text" value="błąd kompilacji"/>
<code>0.1 + 0.2 == 0.3</code>	<input type="text" value="błąd kompilacji"/>
<code>(true && false)</code>	<input type="text" value="błąd kompilacji"/>

Odpowiedź: 1 - błąd 2 - błąd 3 - true 4 - false 5 - false 6 - false

Określ wartość wyrażenia logicznego

Określ wartość wyrażenia logicznego zapisanego w C#. Jeśli jest ono źle skonstruowane, wybierz `błąd kompilacji`.

<code>2f + 2d == 4</code>	<input type="text" value="błąd kompilacji"/>
<code>(false == false)</code>	<input type="text" value="true"/>
<code>(new int()) is ValueType</code>	<input type="text" value="true"/>
<code>"1" + '' == "1"</code>	<input type="text" value="błąd kompilacji"/>
<code>new int() is null</code>	<input type="text" value="błąd kompilacji"/>
<code>(new int()) == 0</code>	<input type="text" value="błąd kompilacji"/>

Odpowiedź: 1 - true 2 - true 3 - true 4 - false 5 - błąd 6 - true ???

Określ wartość wyrażenia logicznego

Określ wartość wyrażenia logicznego zapisanego w C#. Jeśli jest ono źle skonstruowane, wybierz `błąd kompilacji`.

<code>'1' + '2' < 4</code>	<input type="text" value="false"/>
<code>(false == false)</code>	<input type="text" value="true"/>
<code>!!true</code>	<input type="text" value="true"/>
<code>new int[2] == new int[2]</code>	<input type="text" value="false"/>
<code>(new int()) is Stuct</code>	<input type="text" value="błąd kompilacji"/>
<code>(!true)</code>	<input type="text" value="false"/>

Odpowiedzi: 1 - false, 2 - true, 3 - true, 4 - false, 5 - błąd?, 6 - false

Przeciągnij

Liczba pierwsza to taka

Liczba pierwsza to taka liczba naturalna większa od 1, która dzieli się tylko przez 1 i samą siebie. Oto kilka początkowych liczb pierwszych:

2, 3, 5, 7, 11, 13, 17, 19, ...

Aby określić, czy dana liczba jest pierwsza należy zbadać jej dzielniki. Dla zadanej liczby n sprawdzamy kolejne liczby naturalne mniejsze od niej. Jeśli któraś z tych liczb jest dzielnikiem n , oznacza to, że n nie jest liczbą pierwszą.

Algorytm ten można zoptymalizować - wystarczy sprawdzać liczby z przedziału $[2, \sqrt{n}]$.

Napisz w C# funkcję o nazwie `JestPierwsza`, która dla zadanej wartości `n` typu `int` zwróci prawdę, gdy `n` jest liczbą pierwszą oraz fałsz w przeciwnym przypadku.

Wykorzystaj podany powyżej pomysł algorytmu zoptymalizowanego oraz podany poniżej szkielet funkcji, przeciągając i upuszczając odpowiednie bloki kodu w odpowiednie miejsca.

Dane bloki kodu mogą być wykorzystane raz, wiele razy lub ani razu. Wszystkie puste pola w kodzie algorytmu muszą mieć przypisane właściwe bloki kodu.

```
public static  JestPierwsza( int n )
{
    if( n<=1 ) throw new ArgumentOutOfRangeException();
     ;
    while(  )
    {
        if(  ) return  ;
         ;
    }
    return  ;
}
```

true

`n % i > 0`

`n % i == 0`

bool

`n >= i*i`

`int i = 1`

false

`i++`

`i*i <= n`

`i % n > 0`

int

`n / i == 0`

`i % n == 0`

`i >= 2 && i < Math.Sqrt(n)`

`int i = 2`

Odpowiedź:

1) bool	
2) int i = 2	
3) <code>i * i <= n</code>	
4) <code>n % i == 0</code>	5) false
6) <code>i++</code>	
7) true	

Jaki będzie stan końcowy stosu

Jaki będzie stan końcowy stosu (zmienna **S**) oraz kolejki (zmienna **Q**) po wykonaniu następującego fragmentu programu napisanego w C#:

```
int[] a = new int[] { 1, 2, 3, 4 };
int[] b = new int[] { 5, 6, 7, 8 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek());
S.Push(Q.Peek()); Q.Dequeue();
Q.Enqueue(S.Peek()); S.Pop();
Q.Enqueue(S.Peek());
S.Push(Q.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy **Stack** (stos): **Push** - dodaj (wstaw na stos), **Peek** - odczytaj element wierzchołkowy (do usunięcia), **Pop** - usuń (zdejmij) element wierzchołkowy,
- metody klasy **Queue** (kolejka): **Enqueue** - wstaw do kolejki, **Dequeue** - usuń z kolejki, **Peek** - odczytaj element pierwszy (do usunięcia).

Przeciągnij właściwe markery w odpowiednie miejsca. W pustych, nie wypełnionych komórkach, umieść marker z kreską [-].

Jeśli uruchomienie kodu spowoduje pojawienie się wyjątku (stos pusty, kolejka pusta) - we wszystkich komórkach kolejki i stosu umieść marker z kreską [-].

- Stos od wierzchołka:

--	--	--	--	--	--	--	--
- Kolejka od początku:

--	--	--	--	--	--	--	--

1	2	3	4	5	6	7	8	-
---	---	---	---	---	---	---	---	---

Odpowiedzi: stos: 65321???, kolejka:78455 ???

Jaki będzie stan końcowy stosu

Jaki będzie stan końcowy stosu (zmienna `s`) oraz kolejki (zmienna `q`) po wykonaniu następującego fragmentu programu napisanego w C#:

```
int[] a = new int[] { 1, 2, 3, 4 };
int[] b = new int[] { 5, 6, 7, 8 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

Q.Enqueue(Q.Peek()); S.Pop();
S.Push(S.Peek());
S.Push(S.Peek());
Q.Enqueue(Q.Peek()); S.Pop();
S.Push(S.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy `Stack` (stos): `Push` - dodaj (wstaw na stos), `Peek` - odczytaj element wierzchołkowy (do usunięcia), `Pop` - usuń (zdejmij) element wierzchołkowy,
- metody klasy `Queue` (kolejka): `Enqueue` - wstaw do kolejki, `Dequeue` - usuń z kolejki, `Peek` - odczytaj element pierwszy (do usunięcia).

Przeciągnij właściwe markery w odpowiednie miejsca. W pustych, nie wypełnionych komórkach, umieść marker z kreską [-].

Jeśli uruchomienie kodu spowoduje pojawienie się wyjątku (stos pusty, kolejka pusta) - we wszystkich komórkach kolejki i stosu umieść marker z kreską [-].

- Stos od wierzchołka:

3	3	3	2	1	-	-	-
---	---	---	---	---	---	---	---
- Kolejka od początku:

6	7	8	5	5	-	-	-
---	---	---	---	---	---	---	---

1	2	3	4	5	6	7	8	-
---	---	---	---	---	---	---	---	---

Odpowiedzi: stos: 33321???, kolejka: 67855???

Jaki będzie stan końcowy stosu

Jaki będzie stan końcowy stosu (zmienna `s`) oraz kolejki (zmienna `q`) po wykonaniu następującego fragmentu programu napisanego w C#:

```
int[] a = new int[] { 1, 2, 3, 4 };
int[] b = new int[] { 5, 6, 7, 8 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek()); Q.Dequeue();
S.Push(Q.Peek()); Q.Dequeue();
Q.Enqueue(S.Peek()); S.Pop();
Q.Enqueue(S.Peek()); S.Pop();
S.Push(Q.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy `Stack` (stos): `Push` - dodaj (wstaw na stos), `Peek` - odczytaj element wierzchołkowy (do usunięcia), `Pop` - usuń (zdejmij) element wierzchołkowy,
- metody klasy `Queue` (kolejka): `Enqueue` - wstaw do kolejki, `Dequeue` - usuń z kolejki, `Peek` - odczytaj element pierwszy (do usunięcia).

Przeciągnij właściwe markery w odpowiednie miejsca. W pustych, nie wypełnionych komórkach, umieść marker z kreską [-].

Jeśli uruchomienie kodu spowoduje pojawienie się wyjątku (stos pusty, kolejka pusta) - we wszystkich komórkach kolejki i stosu umieść marker z kreską [-].

- Stos od wierzchołka:

7	3	2	1	-	-	-
---	---	---	---	---	---	---
- Kolejka od początku:

8	4	6	5	-	-	-
---	---	---	---	---	---	---

odp: stos 7321; kolejka 8465

Jaki będzie stan końcowy stosu

Jaki będzie stan końcowy stosu (zmienna `S`) oraz kolejki (zmienna `Q`) po wykonaniu następującego fragmentu programu napisanego w C#:

```
int[] a = new int[] { 1, 2, 3, 4 };
int[] b = new int[] { 5, 6, 7, 8 };

Stack<int> S = new Stack<int>(a);
Queue<int> Q = new Queue<int>(b);

Q.Enqueue(Q.Peek()); S.Pop();
S.Push(S.Peek());
S.Push(S.Peek());
Q.Enqueue(Q.Peek()); S.Pop();
S.Push(S.Peek()); Q.Dequeue();
```

Wyjaśnienia:

- metody klasy `Stack` (stos): `Push` - dodaj (wstaw na stos), `Peek` - odczytaj element wierzchołkowy (do usunięcia), `Pop` - usuń (zdejmij) element wierzchołkowy,
- metody klasy `Queue` (kolejka): `Enqueue` - wstaw do kolejki, `Dequeue` - usuń z kolejki, `Peek` - odczytaj element pierwszy (do usunięcia).

Przeciągnij właściwe markery w odpowiednie miejsca. W pustych, nie wypełnionych komórkach, umieść marker z kreską [-].

Jeśli uruchomienie kodu spowoduje pojawienie się wyjątku (stos pusty, kolejka pusta) - we wszystkich komórkach kolejki i stosu umieść marker z kreską [-].

- Stos od wierzchołka:

3	3	3	2	1	-	-	-
---	---	---	---	---	---	---	---
- Kolejka od początku:

6	7	8	5	5	-	-	-
---	---	---	---	---	---	---	---

1	2	3	4	5	6	7	8	-
---	---	---	---	---	---	---	---	---

Odpowiedź: ???

Dla jakiej wartości początkowej zmiennej

Dla jakiej wartości początkowej zmiennej całkowitej `x` program wypisze 8-krotnie gwiazdkę `*`.

Program zapisany jest w pseudokodzie.

```
x ← ???
while x ≤ 7 do
{
  pisz("*")
  x ← x + 1
}
```

Podaj wartość zmiennej `x`

Podaj 8-bitowy

Podaj 8-bitowy kod zapisanej w systemie dziesiętnym liczby $(-13)_{10}$ w kodowaniu znak-moduł.

Odpowiedź:

Dla podanego fragmentu kodu

Dla podanego fragmentu kodu:

```
if( temperatura > gornyLimit ) {  
    if( niebezpieczenstwo ) wywolajAlarm();  
} else  
    wlaczReaktor();
```

- Procedura `wywolajAlarm()` zostanie uruchomiona, gdy temperatura przekroczy górny limit i wystąpi niebezpieczeństwo
- Procedura `wlaczReaktor()` zostanie uruchomiona, gdy temperatura nie przekroczy górnego limitu , bez względu na niebezpieczeństwo

uzupełnij zdania tak, aby były prawdziwe (przeciągnij i upuść z odpowiednich grup).

bez względu na wartość temperatury	gdy temperatura przekroczy górny limit	gdy temperatura nie przekroczy górnego limitu
lub nie będzie niebezpieczeństwa	lub wystąpi niebezpieczeństwo	i nie będzie niebezpieczeństwa
i wystąpi niebezpieczeństwo	, bez względu na niebezpieczeństwo	

czy podany zapis

Czy podany zapis jest kompletnym i właściwym komentarzem?

```
/* // */
```

Wybierz jedną, najlepiej wyjaśniającą odpowiedź.

Wybierz jedną odpowiedź:

- ☒ To jest właściwy sposób komentowania. Fragment `//` jest ignorowany przez kompilator.
- ☐ Taka kombinacja komentarzy jest dozwolona, ale nie zalecana. Kompilator skompiluje kod, ale zgłosi ostrzeżenie (ang. *warning*) sugerując eliminację zagnieżdżonych komentarzy.
- ☐ Takia kombinacja komentarzy blokowego i liniowego jest niedozwolona i kompilator zgłosi błąd.
- ☐ Nie, komentarz blokowy `/* ... */` nie będzie uznany za zamknięty. Fragment `/*...//` zostanie uznany przez kompilator za komentarz, ale w konsekwencji fragment `... */` nie będzie miał odpowiadającego mu znacznika otwarcia komentarza.