Programowanie wstęp

Wyświetlanie

1. Napisz program, który wyświetli napis “Witaj Świecie!”

2. Napisz program, który wyświetli w konsoli:

   Zaczynam naukę na kursie: “Java od Podstaw”
   Pora więc na podstawy

   J
   A
   V
   Y

3. Napisz program, który wyświetli w konsoli:

   "Wyrażenia" generują nowe dane.

   "Zmienne" przechowują dane.

   Operacje "Input\output" umożliwia interakcję z użytkownikiem.

   "Instrukcje warunkowe" sterują przebiegiem programu.

Literały i wyrażenia

4. Wykonaj poniższe operacje wyświetlając jednocześnie ich wyniki:

   - dodaj dwie liczby całkowite
   - dodaj tekst z liczbą całkowitą
   - podziel liczbę całkowitą przez dziesiętną
   - wykonaj sprawdzenie logiczne (np. coś większe od czegoś)

5. Ćwiczenia prostych wyrażeń:

   a. Wykorzystaj javę do obliczenia sumy z równania:
      5*10
      oczekiwany wydruk: 50

   b. Wykorzystaj javę do obliczenia sumy z równania:
      $\frac{10+20}{7}$

oczekiwany wydruk: 4.28571428571

c. Przerób program na taki, który informuje o wszystkich krokach które są wykonywane np:
10 + 20 to 30
30 / 7 daje 4.28571428571

d. Sprawdź za pomocą javy czy wynik podnoszenia 15 do 2 potęgi jest większy od 100
Oczekiwany wydruk:

czy wynik równania jest większy od 100: true

Zmienne

6. W nowej klasie Kalkulator zadeklaruj 3 zmienne oraz zainicjalizuj je wartościami: 20, 5 oraz 12. Zadeklaruj jeszcze jedną zmienną, wstawiając do niej sumę wcześniej przygotowanych zmiennych. Program ma wyświetlić: “Podane liczby to: …, …, … a ich suma wynosi: ...!”. W miejscach “...” powinieneś odnosić się do wcześniej przygotowanych zmiennych aby program mógł wypakować z nich wartości i je wyświetlić.

7. Napisz program, w którym zadeklarujesz zmienną z ilością minut. Program ma obliczyć ile to godzin oraz ile to sekund.

    - **wariant trudniejszy:** wykonaj ten sam program posługując się tylko jedną zmienną i zmieniając jej wartość przy każdej operacji

8. Napisz prosty kalkulator walut, w którym zadeklarujsz zmienne z kwotą w złotówkach, aktualnym kursem dolara oraz kursem euro. Program ma wyświetlić kwotę po przeliczeniu na dolary oraz po przeliczeniu na euro.

9. Napisz program który oblicza wskaźnik BMI: wzór: waga/wzrost². Waga ma być podana w kilogramach, a wzrost w metrach. Dodatkowo program ma sprawdzić czy BMI jest powyżej 24,9 - nadwaga lub czy BMI jest poniżej 18,5 - niedowaga. Poinformuj w konsoli o każdym działaniu programu.

Scanner i Random

10. Napisz program który: pobierze imię, nazwisko i wiek, następnie przywita użytkownika oraz poinformuje czy użytkownik jest pełnoletni.

    Przykładowy rezultat:
    Cześć, jak masz na imię?

Jan
Jak masz na nazwisko?
Kowalski
Cześć Jan! Proszę podaj jeszcze swój wiek:
30
Już wiem czy jesteś pełnoletni: true

- **wariant trudniejszy:** program ma sprawdzić czy wiek podany przez użytkownika jest poprawną wartością (między 0 a 100) oraz do jakiej kategorii wiekowej się kwalifikuje - dziecko, nastolatek, osoba dorosła lub wiek emerytalny.

11. Napisz program który wylosuje:
    a. Liczbę z przedziału 0 - 10
    b. Liczbę z przedziału 1-6
    c. Liczbę dziesiętną z przedziału 0.0 do 1.0
    d. Liczbę dziesiętną z przedziału 1.0 do 100.0
    e. true lub false

12. Napisz program, który będzie generował losowe liczby, tak, jakbyś rzucał kostkami do gry. Napierw ma zapytać ile ścianek ma kostka, którą ma rzucić, a następnie zasymuluje rzut kostką o wybranej liczbie ścian i wyświetli wylosowaną liczbę.

    Przykładowy rezultat:
    Program rzuci dla Ciebie kostką do gry i wylosuje liczbę!
    Proszę, podaj ile ścianek ma mieć kostka która rzucasz? 4, 6, 10 a może 20?
    6
    Rzucam kostką o ilości ścianek: 6
    Wyszło mi: 4

    - **wariant trudniejszy:** pobierz od użytkownika jaka ma być najmniejsza i jaka największa wartość losowania a następnie wylosuj 5 liczb z zadanego zakresu.

Instrukcje warunkowe

13. Pobierz z konsoli liczbę.
    a. wyświetl czy jest to liczba dodatnia, ujemna lub równa 0
    b. wyświetl czy jest to liczba parzysta czy nieparzysta

    - **wariant trudniejszy:** pobierz trzy liczby z konsoli, wyświetl największą z nich

14. Napisz program, który zapyta użytkownika czy woli programować, grać w gry czy może jedno i drugie.

Jeśli użytkownik wybierze programowanie, program poprosi o podanie ulubionego języka programowania. Jeśli użytkownik wybierze granie w gry, poprosi o podanie ulubionej gry. Jeśli użytkownik wybierze jedno i drugie, zada oba pytania.

Na koniec program powinien wygenerować odpowiedni komunikat używając danych podanych przez użytkownika.

Przykładowy przebieg programu:

Hej opowiedz mi o swoich zainteresowaniach, wybierz jedną z 3 opcji:
1. *Lubię programować*
2. *Lubię grać w gry*
3. *Lubię programować oraz grać w gry*

3
O, lubisz jedno i drugie.
W jakim języku lubisz programować?
java
W jaką grę lubisz grać?
Starcraft 2
Pora na małe podsumowanie:
Twój ulubiony język programowania to java.
Twoja ulubiona gra to Starcraft 2.
Zapamiętam!

(jeśli chcesz możesz zmienić temat pytań)

15. Zadeklaruj dwie zmienne boolean “deszcz” oraz “swieciSlonce”.

    Program ma wyświetlić pogodę sprawdzając ich zawartość:

    - jeśli pada i świeci słońce: tęcza!
    - jeśli nie pada i świeci słońce: jest pogodnie!
    - jeśli nie pada i nie świeci słońce: jest pochmurno..
    - jeśli pada i nie świeci słońce: ulewa!

    - **wariant trudniejszy:** przygotuj wcześniejszą wersję programu ale zawartość zmiennych deszcz oraz swieciSlonce określaj dynamicznie, pytając użytkownika za pomocą scannera. Zinterpretuj jego odpowiedzi uzupełniając zmienne wartościami true lub false.

16. W sklepie ze sprzętem AGD oferowana jest sprzedaż ratalna.

Napisz program umożliwiający wyliczenie wysokości miesięcznej raty za zakupiony sprzęt. Danymi wejściowymi dla programu są:

• cena towaru (od 100 zł do 10 tys. zł),
• liczba rat (od 6 do 48).

Program powinien sprawdzać, czy podane dane mieszczą się w określonych powyżej zakresach, a w przypadku błędu wyświetlać stosowny komunikat i natychmiast zakończyć działanie programu. (użyj instrukcji return)
Kredyt jest oprocentowany w zależności od liczby rat:

- od 6–12 wynosi 2.5% ,
- od 13–24 wynosi 5%,
- od 25–48 wynosi 10%.

Obliczona miesięczna rata powinna zawierać również odsetki.

> **Podpowiedź**: możesz zakończyć metodę main i tym samym wyjść z programu w dowolnym momencie stosując instrukcję: return; lub System.exit(int);

- **wariant trudniejszy:** zadbaj o jak najmniejszą ilość powtórzeń w kodzie

Projekty

Gra w kości

Przygotuj program który pozwoli w konsoli zagrać w następującą grę:

https://iftemplar.github.io/works/pig-dice-game/

Zagraj w tę grę i postaraj się zrozumieć wszystkie jej zasady.

Pierwsza wersja gry ma trwać tylko 3 tury oraz obejmować tylko jednego gracza i nie będzie w niej żadnego szczególnego działania przy wyrzuceniu 1.

Przykładowe działanie:

Chcesz rzucić kostką czy spasować?
Twój wynik to: 0

1. *rzucam*
2. *pasuję*

1
Rzucasz… wypadło 5.

Chcesz rzucić kostką czy spasować?
Twój wynik to: 5

3. *rzucam*
4. *pasuję*

1
Rzucasz… wypadło 1.

Chcesz rzucić kostką czy spasować?

Twój wynik to: 6

5. *rzucam*
6. *pasuję*

1

Rzucasz… wypadło 3

Koniec gry, tym razem uzbierałeś 9 punktów!

https://iftemplar.github.io/works/pig-dice-game/

Wersja 2:

- jeśli wypadnie 1 to przegrywasz
- gra trwa tak długo dopóki nie uzyskasz 20 punktów (w finalnej wersji można zwiększyć ten limit, 20 jest po to żeby szybko można było testować)

Wersja 3:

- istnieją 2 pule punktów trwała i tymczasowa
- rzuty trafiają do tymczasowej
- spasowanie dodaje punkty z tymczasowej do trwałej i zeruje tymczasową
- jeśli wypadnie 1 do tymczasowa jest zerowana

Wersja 4:

- po spasowaniu i po wyrzuceniu 1 następuje tura drugiego gracza

Gra quiz

Stwórz quiz z pytaniami na temat jakiegoś hobby np. podróże

Wersja 1:

- Quiz to 3 pytania z krótką odpowiedzią na pytanie np. jaka jest stolica Hiszpanii?
- Program drukuje na koniec wynik

Wersja 2:

- Zmodyfikuj quiz aby były wyłącznie pytania jednokrotnego wyboru

Wersja 3:

- Przygotuj program na to, aby był w stanie obsłużyć 10 pytań.
- Spraw aby wynik punktowy był zależny od czasu

Wersja 4:

- Zmodyfikuj program aby losował x pytań z puli w której mogą być zarówno pytania otwarte jak i jednokrotnego wyboru

Wersja 5:

- Dodaj obsługę pytań wielokrotnego wyboru

Wirtualny stworek

Przygotuj program w którym użytkownik będzie opiekował się wirtualnym stworzeniem.

Wersja 1:

- Opiekun może nadać stworkowi imię
- Program losuje stworkowi statystyki takie jak:
  - zdrowie
  - najedzenie
  - chęć zabawy
- Program wydrukuje stan stworka i zapyta co chcemy z nim zrobić. Przykładowe działanie

  Jak ma się nazywać Twój mały dinozaur?

  Zaur

  Właściciel Zaur miał wypadek, od teraz Ty będziesz się nim zajmował.

  Stan dinozaura:
  - *w dobrej kondycji*
  - *głodny*
  - *znudzony*

  Co chcesz zrobić?
  1. *położyć go do snu (zregeneruje zdrowie)*
  2. *nakarm*
  3. *baw się*

…

Wersja 2:

- Program poprosi o decyzję trzy razy
- Dodaj kilka ciekawych zależności np.
  - jeśli nie jest głodny nie będzie chciał jeść
  - jeśli jest bardzo głodny a go nie nakarmimy spadnie mu poziom zdrowia
  - jeśli jest bardzo głodny i nudzi się to i tak nie będzie chciał się bawić bo nie ma nastroju

Wersja 3:

- Każda tura z decyzją gracza to jeden dzień. Cel gry jest taki aby po 18 dniach stworek był w jak najlepszej kondycji
- Po drodze może umrzeć jeśli nie dba się o niego wystarczająco dobrze
- Na koniec gry program oblicza wartość punktową zależną od tego jak dobrze nam poszło

Wersja 4:

- Stworka można karmić jednym z 3 rodzajów jedzenia
- Każdy rodzaj jedzenia jest pod jakimś względem inny np. mniej sycący ale zdrowy więc regeneruje zdrowie

Wersja 5:

- Program przy każdym uruchomieniu generuje inne 3 rodzaje specjalnego jedzenia z większej puli lub stosując losowanie

Program można rozbudowywać jeszcze na milion różnych sposobów więc istnieje duża dowolność w implementacji projektu. Celem jest stworzenie jak najbardziej rozbudowanej i ciekawej symulacji.

Sklep z artefaktami

Przygotuj program który będzie prostą grą symulującą pracę sprzedawcy magicznych artefaktów w fantastycznym świecie. Grasz sprzedawcą, przychodzą do Ciebie bohaterowie kupować sprzęt który będzie potrzebny do walki z potworem.

wersja 1:

- program tworzy bohatera z losową obroną i siłą
- Ustaw losowanie tak aby każdy z tych parametrów mógł przyjmować wartości od 0-4 ale ich suma zawsze wynosiła 4
- heros posiada 25 monet
- program losuje przedmiot u sprzedawcy
- przedmiot ma swój losowy koszt 5, 10 lub 15 monet
- przedmiot ma losowe statystyki (siła i obrona) i ich łączna wartość wynosi 1 za każde 5 monet ceny przedmiotu
- Na tym etapie program nie oferuje interakcji

Przykładowy wydruk programu:

Do sklepu przychodzi heros:
Imię: Joanna Odważna
Siła: 3
Obrona: 1
Monety: 25

Przedmiot na sprzedaż:
Nazwa: srebrna zbroja
Siła: 2
Obrona: 1
Cena: 15

Wersja 2:

- Po tym jak program wygeneruje przedmiot gracz podejmuje decyzje czy sprzedać ten przedmiot czy nie
- Postać musi mieć dość monet
- Po sprzedaży ilość monet herosa spada a statystyki rosną
- Program generuje jeszcze 2 kolejne przedmioty które można sprzedać lub nie
- Po zakończonych transakcjach heros walczy z potworem o losowych statystykach
- Do generowania statystyk potwora wykorzystaj podobny kod jak przy herosie ale potwór powinien być odpowiednio silniejszy
- Jeśli heros ma co najmniej taką siłę jak obrona potwora - pokonuje potwora
- Jeśli potwór ma większą siłę niż obrona herosa - heros ginie
- gracz dostaje 1 punkt za każde 10 monet które zarobił, 1 punkt jeśli heros przeżył i 1 punkt jeśli pokonał potwora

Pomysły na dalszy rozwój:

- Gra toczy się dalej do pokonania 3 potworów, każdy kolejny potwór jest silniejszy i zostawia po pokonaniu złoto dla wygranego herosa, heros wracając do sklepu ma znowu bazowe statystyki
- Gracz wybiera z puli 5 przygotowanych przedmiotów zamiast być pytany o każdy jeden po drugim
- Przedmioty i herosi mają swój typ - mag, wojownik, łotr i kleryk. Jeden przedmiot jak i jeden kleryk może mieć kilka różnych typów. Sprzedaż jest możliwa tylko gdy chociaż jeden z typów przedmiotu zgadza się z typem herosa.

Ciekawostka: zadanie jest wzorowane na grze karcianej Bargain Quest

Dodatkowe zadania:

Pętle while

17. Przygotuj pętle, w której będziesz losować liczbę od 1 do 10 oraz wyświetlać ją. Jeśli pętla wylosuje liczbę 10 ma zostać przerwana i zakomunikować koniec programu.

Przykład działania:

Zaczynamy losowanie!
3
6
3
2
10
Trafione!

- **wariant trudniejszy:** program na koniec wyświetla po ilu próbach udało jej się wylosować liczbę a następnie losuje liczby jeszcze raz dokładnie tyle razy.

18. Wykonaj program, który pomoże Ci zliczać punkty uzyskanie na koniec rundy w skomplikowanych grach planszówkowych :) Użytkownik ma mieć możliwość podawania dowolnej ilości liczb. Program ma je sumować na bieżąco, a jeśli użytkownik wpisze 0 to znaczy, że skończył więc program ma ogłosić łączną sumę punktów.

- **wariant trudniejszy:** wykonaj poniższy program ale tym razem instrukcją kończącą wprowadzanie liczb ma być liczba -1. Zadbaj o to aby program dalej zliczał liczby poprawnie.

Przykład działania:

Wprowadzaj uzyskane punkty cząstkowe. Aby zakończyć wpisz 0. Policzę dla Ciebie ich sumę :)

Wprowadź liczbę
4
wprowadź liczbę
1
wprowadź liczbę
10
wprowadź liczbę
0

Uzyskane punkty: 15, gratulacje!

19. Wykonaj pętle która wyświetli:

    a) liczby rosnąco od 1 do 10,
    b) liczby malejąco od 99 do 66,
    c) wyłącznie parzyste liczby od 20 do 40,
    d) 5 razy Twoje imię.

    - **wariant trudniejszy:** wróć do zadania z autobusem
    - W zadaniu z przeprowadź 20 symulacji i policzyć ile razy użytkownik był łapany przez kontrolera a ile razy mu się upiekło. Możesz też wyliczyć “opłacalność” jazdy bez biletu czyli ile zaoszczędzi się na bilecie a ile straci na karach przy określonym prawdopodobieństwie spotkania kontrolera.

20. Napisz program, który wyświetla tajny komunikat. Zanim wyświetli komunikat prosi o podanie hasła liczbowego (sam wymyśl jakie ma być hasło). W przypadku wprowadzenia niepoprawnego hasła program pyta ponownie o liczbę tak długo dopóki użytkownik nie wpisze poprawnego hasła.

    - **wariant trudniejszy:** wróć do zadania z BMI.
    - W zadaniu z BMI zastosuj pętle while lub do-while aby wczytywać wzrost i wagę tak długo dopóki nie zostaną podane sensowne wartości. Tzn. program nie powinien pozwolić na ustawienie wzrostu 10 metrów.

21. Zrób grę w zgadywanie, w której program losuje liczbę od 1 do 10, a gracz ma próbować ją odgadnąć. W przypadku każdej odpowiedzi gracz dostaje informację zwrotną: za wysoko/za nisko/trafiona. Jeśli zgadnie program ma zakończyć działanie i wyświetlić po ilu próbach udało mu się odgadnąć liczbę.

- **wariant trudniejszy:** Ulepsz grę aby pytała tylko raz jeśli od razu trafisz i mówiła dodatkowo: “gorąco!” jeśli byłeś blisko oraz “zimno…” jeśli byłeś daleko.

Tablice

22. Napisz program “Poszukiwany”, stwórz i przypisz do odpowiednich zmiennych dwie tablice:
    a) tablica liczb z 4 miejscami,
    b) tablica tekstów z 2 miejscami.

Następnie przypisz ręcznie:

a) do liczb: osobno dzień, miesiąc, rok ucieczki z więzienia,
b) na 4 pozycji wpisz kwotę nagrody,
c) do tekstów: imię i nazwisko uciekiniera.

Wyświetl uzupełnioną wiadomość:

a) “ … - … - ….. zbiegł więzień …… …….. !”
b) “Nagroda za przyprowadzenie: …….$!”

23. Stwórz skróconym sposobem tablicę:
    a) liczb zmiennoprzecinkowych z 4 elementami,
    b) wartości boolean z 2 elementami
    c) tablicę znaków z literami: j, a, v, a

Wyświetl zawartość każdej z tablic pomagając sobie metodą toString(...) z biblioteki Arrays.

- **wariant trudniejszy:** stwórz tablicę obiektów Random uzupełnioną trzema obiektami. Za pomocą drugiego z Randomów wylosuj i wydrukuj liczbę od 0 do 5.

24. Szybkim sposobem stwórz tablicę 10 liczb i wstaw wartości z przedziału 1 - 1000.
    a) za pomocą pętli wyświetl wszystkie liczby,
    b) wyświetl tylko liczby większe niż 100,
    c) wyświetl tylko nieparzyste liczby i mniejsze niż 750.

- **wariant trudniejszy:** za pomocą pętli policz średnią tych liczb

25. Napisz program, który zapyta użytkownika ile notatek potrzebuje i stworzy tablicę tekstów o wpisanej wielkości. Następnie stwórz pętlę, która pozwoli wprowadzać notatki tyle razy ile jest miejsc w tablicy i wyświetl wszystkie pozycje tablicy.

- **Q&A:**
- Q: Dlaczego scanner nie wczytuje tekstu mimo, że wywołałem instrukcję nextLine?
- A: Scanner posiada bufor w którym zapisuje pobrane informacje. Metoda pobierająca liczbę czyta wyłącznie wartość liczbową, ale użytkownik w konsoli przekazuje nie tylko ją, ale także separator powstający przy wciśnięciu klawisza “enter”. Następne wywołanie nextLine()

pobierze tekst aż do separatora włącznie. Pamiętaj jednak, że Scanner w pierwszej kolejności sprawdzi swój bufor i jeśli będzie on pusty to dopiero wtedy pobierze dane z konsoli. W związku z tym, jeśli wcześniej wykorzystaliśmy metodę pobierającą liczbę, przed metodą pobierającą tekst należy najpierw oczyścić bufor scannera dodatkowym wywołaniem “nextLine()”. Można też po prostu zrobić nowy skaner.

- **wariant trudniejszy:** Na koniec program ma losować jedną z notatek i wyświetlać. Wykorzystaj program do wprowadzenia listy osób z grupy i zobacz kogo wylosujesz. Jeśli chcesz możesz napisać do tej osoby coś miłego na slacku :)

Pętle for

26. Za pomocą pętli for:
    - wyświetl liczby od 0 do 5000.
    - przygotuj tablicę 5 imion i wyświetl
    - wyświetl pierwsze 20 potęg liczby 2:
        0, 2, 4, 8, 16…..

27. Za pomocą pętli **for** wylosuj milion liczb z przedziału 1-100 i zapisz je w tablicy. Następnie stosując **pętle for** each policz ile razy wystąpiła liczba 25 i wyświetl tę informację na końcu programu

    - **wariant trudniejszy:** Program ma zliczyć ilość wystąpień każdej z liczb i wyświetlić efekty na koniec.

28. Przygotuj tablicę 10 liczb z przedziału -100 do 100. Za pomocą pętli for each:

- policz średnią z tych liczb
- wyświetl największą liczbę
- wyświetl najmniejszą liczbę

29. Program pobiera wysokość, szerokość oraz znak wypełnienia kwadratu a następnie za pomocą pętli for drukuje kwadrat w konsoli, o zadanych parametrach.

    Przykład działania:

    Wprowadź znak wypełnienia prostokąta:
    &
    Wprowadź wysokość prostokąta:
    4
    Wprowadź szerokość prostokąta:
    7
    Rysuję!
    &&&&&&&

&&&&&&&
&&&&&&&

30. Napisać program rysujący w konsoli „choinkę” złożoną ze znaków gwiazdki (*). Użytkownik programu powinien podać liczbę całkowitą n, n > 0, określającą wysokość choinki (liczbę wierszy).

    Przykład: dla n = 5 wynik powinien wyglądać następująco:

    *
    **
    ***
    ****
    *****

    Opcjonalny wariant dla ambitnych: program rysuje symetryczną choinkę:

    *
    ***
    *****
    ****
    ****

31. Obejrzyj uważnie ten film: **link** a następnie korzystając z pętli for posortuj tablicę według algorytmu sortowania bombelkowego.

Projekt Loteria

Symulator loterii liczbowej:

1. wprowadzenie 6 liczb (1-24)
2. wylosowanie 6 liczb (1-24)
3. sprawdzenie, które liczby są trafione
4. wyświetlenie wyniku i nagrody:

a) 0-2 = 0zł, 3 = 16 zł
b) 4 = 200zł, 5 = 4 000 zł
c) 6 = 1 500 000 zł

Użyj każdego z poznanych narzędzi: zmienna, operatory, warunek, tablice, Scanner, Random, Pętle..

- **wariant trudniejszy:** Liczby wprowadzane ani liczby losowane nie mogą się powtarzać.

Metody statyczne

32. Przygotuj następujące metody i przetesuj w main:
    a. metoda wyświetla napis “hello”

b. metoda przyjmuje jako parametr imię i drukuje “Cześć <imie>!”
c. metoda przyjmuje tekst oraz liczbę i drukuje ten tekst taką ilość razy
d. metoda przyjmuje dwa parametry imię oraz wiek. Jeśli przekazano niski wiek, wyświetla “cześć <imie>” jeśli powyżej 20, wyświetla “dzień dobry <imie>”
e. metoda przyjmuje tablicę imion i wita wszystkie przekazane osoby :)

- **wariant trudniejszy:** metoda przyjmuje ilość sztuk towaru, cenę oraz podatek. Powinna wyliczyć łączoną cenę towaru. Jeśli cena przekracza 100 zł powinna ustawić zmienną boolean darmowaWysylka na true. Przygotuj drugą, pomocniczą metodę dla tej pierwszej, która przyjmie łączoną cenę towaru oraz informację o darmowejWysylce i wyświetli stosowny komunikat o cenie do zapłaty oraz informację czy przesyłka będzie za darmo. Do wyświetlonej ceny powinna dodać ew. koszt wysyłki + 10zł.

33. Przygotuj następujące metody i przetesuj w main:
    a. metoda zwraca Twoją ulubioną liczbę. Przyjmij do zmiennej w main i wyświetl wartość zwróconą
    b. metoda przyjmuje trzy liczby i zwraca ich sumę
    c. metoda losuje liczbę od 1 do 10 i zwraca
    d. metoda przyjmuje ilość lat i zwraca ilość dni
    e. metoda przyjmuje dwie liczby i dzieli pierwszą przez drugą. Jeśli druga liczba to 0 metoda ma zwrócić -1 aby zakomunikować błąd.
    f. metoda przyjmuje dzień, miesiąc i rok urodzenia i zwraca je w postaci String formatując w ten sposób: “RRRR-MM-DD”
    g. metoda przyjmuje trzy liczby, zwraca true jeśli wszystkie trzy liczby są identyczne. W main, testując metodę wyświetl “Takie same” jeśli zwróciła true oraz “Inne” jeśli false.
    h. metoda przyjmuje tablicę liczb i zwraca ich sumę
    i. metoda przyjmuje tablicę liczb oraz liczbę całkowitą, zwraca true jeśli w tablicy istnieje liczba większa od przekazanej, false jeśli nie

- **wariant trudniejszy:** metoda przyjmuje liczbę całkowitą. Powinna zwrócić tablicę liczb pierwszych nie większych niż przekazana liczba.

Rzutowanie

34. Przygotuj następujące metody i przetesuj w main:
    a. metoda przyjmuje dwa argumenty typu double oraz int. Zwraca pierwszą liczbę podniesioną do potęgi wartości drugiej liczby. Wynik zwraca w double
    b. przetestuj powyższą metodę dla liczb 3 oraz 5, wynik zapisz do zmiennej int i wyświetl. Wykorzystaj rzutowanie. Po przekazaniu jakich wartości program będzie wyświetlać nieprawidłowe wyniki? Przetestuj to.
    c. przygotuj metodę przyjmującą ilość minut i zwracającą ilość milisekund. Jako typ zwracany metody zadeklaruj wartość long. Wykorzystaj tę metodę dla przeliczenia 5 min. Wartość zwróconą przypisz do zmiennej int i wyświetl. Wykorzystaj rzutowanie. Potestuj metodę i sprawdź, jakie argumenty musiałbyś przekazać aby metoda zaczęła dawać niepoprawne wyniki?

- **wariant trudniejszy:** stwórz metodę która przyjmuje liczbę int oraz zwraca informację ile razy rzutowanie tej liczby na typ byte przekroczy jego maksymalną wartość.

Metody klasy String

35. Stwórz zmienną typu String i zainicjuj ją dowolnym zdaniem złożonym, a następnie:
    a) wyświetl zdanie dużymi literami,
    b) zapisz w zmiennej odpowiedniego typu ilość symboli w zdaniu,
    c) wyświetl pierwszą literę w tym zdaniu,
    d) nadpisz oryginalną zmienną tak, żeby zawierała zdanie wielkimi literami,
    e) wyświetl odpowiedź na pytanie czy w zdaniu znajduje się słowo “nie” (true/false),
    f) wyświetl fragment tekstu od znaku 5 do 13,
    g) wyświetl fragment tekstu od znaku 7 do końca,

Podpowiedź: możesz wykorzystać metody: repeat(int), charAt(int), substring(int,int), substring(int), contains(String), toUpperCase(), length()
Podpowiedź 2: skrót do wyświetlenia dokumentacji metody to ctrl + q (Windows)

- **wariant trudniejszy:**
    h) wyświetl ostatnią literę zdania, program ma działać poprawnie dla zdania dowolnej długości
    i) wyświetl zdanie 5 razy dodając po drodze symbol nowej linii metodą concat(String)

Podpowiedź 3: możesz wywoływać metody wielokrotnie o ile produktem metody jest nowa wartość typu referencyjnego np: tekst.toUpperCase().charAt(0)

Biblioteki statyczne

36. Wykonaj następujące zadania:
    a. Wyświetl piątą potęgę liczby 2 wykorzystując bibliotekę Math.
    b. stwórz metodę która przyjmie tekst i zwróci true jeśli ostatni znak będzie cyfrą. Przykładowo “ul. Wiejska 1” -> true

        - **Podpowiedź:** wykorzystaj metody charAt oraz metodę isDigit z biblioteki Character

    c. napisz metodę, w której za pomocą scannera pobierzesz liczbę i ją zwrócisz. Metoda ma zwracać -1 jeśli poda się tekst.

        - **Podpowiedź:** wczytaj liczbę jako tekst, sprawdź czy jest tam na pewno liczba a potem przekonwertuj ją do typu int za pomocą metody parsującej z biblioteki Inteeger

- **wariant trudniejszy:** jeśli poda się litery metoda ma pozwolić na powtórne wprowadzanie danych do skutku.

d. napisz metodę, która wyszukuje element i zwraca jego pozycję. Wykorzystaj algorytm sortowania binarnego zaimlementowany w bibliotece Arrays. Metoda ma najpierw posortować przyjęty zbiór.

Pola statyczne

37. W nowej klasie zadeklaruj pole odpowiedniego typu o nazwie “dna”, a jego treść uzupełnić losową sekwencją 30 liter wykorzystując **ten** generator kodu DNA.

    Stwórz metody, które wykonają następujące operacje:
    a) zapisanie kodu dużymi literami,
    b) zmiana wszystkich A na T i wszystkich G na C,
    c) przycięcie kodu do pierwszych 15 znaków,

    - wariant trudniejszy: wstawienie spacji co trzeci znak.

Podpowiedź: wykorzystaj metody toUpperCase(), replace(char,char), substring(int,int) i charAt(int)

//modyfikatory dostępu

Instrukcje warunkowe switch

38. Stwórz metodę, i przekaż jej jako parametr stronę świata. Metoda ma symulować poruszanie się po wirtualnym świecie w 4 możliwych kierunkach.W mainie zapytaj użytkownika, w którą stronę chce się udać. Może napisać:
    a. północ
    b. południe
    c. wschód
    d. zachód

Zastosuj instrukcję warunkową **switch** aby, w zależności którą stronę wybrał, zwrócić mu inny komunikat Np. “Wyruszyłeś do stolicy przez las na wschodzie, po drodze zostałeś napadnięty przez bandytów, rozpoczyna się walka!”

Podpowiedź: aby przypisać kod do sytuacji w której nie znaleziono pasującej wartości w switchu zastosuj słowo kluczowe “default” zamiast “case”.

39. Napisz metodę liczącą, która przyjmie dwie liczby oraz znak równania (+,-,* lub /). Za pomocą instrukcji switch dobierz do znaku odpowiednią operację na podanych liczbach i zwróć jej wynik. .

40. Metoda przyjmuje w argumencie dzień oraz miesiąc w postaci liczb całkowitych. Na tej podstawie ma zwrócić w postaci String odpowiednią porę roku. Wykorzystaj switch oraz instrukcje warunkowe if.

Pora roku	Początek	Koniec
Wiosna	21 marca	20 czerwca
Lato	21 czerwca	22 września
Jesień	23 września	21 grudnia
Zima	22 grudnia	20 marca

Podpowiedź: postaraj się skrócić kod pomijając instrukcje break; tam gdzie jest to możliwe. W miesiącach na granicy 2 pór roku zastosuj instrukcje warunkowe if.

41. Wykorzystaj mechanizm varargs aby stworzyć metodę, który przyjmie dowolną ilość argumentów liczbowych. Metoda ma zwrócić losową wartość z tego zbioru.

42. Stwórz metodę, która ma pozwolić pobrać liczbę za pomocą scannera. Zastosuj dwa parametry liczbowe aby można było przekazać metodzie informacje jaka jest maksymalna liczba oraz minimalna którą może wpisać użytkownik. Metoda ma pobierać liczbę od użytkownika tak długo, dopóki użytkownik nie wpisze liczby z podanego zakresu. Jeśli poda liczbę nieprawidłową, metodę rekurencyjnie ponownie i zwróć wynik z nowego wywołania.

Projekt Asystent

Stworzymy program demonstrujący Twoje przykładowe aplikacje.

Etap 1:

Stwórz pakiet asystent i skopiuj do niego trzy klasy najciekawszych programów które do tej pory zrobiłeś. W klasie Asystent zaimplementuj metodę główną void rozpocznijProgram( ) i wywołaj ją w metodzie main. Następnie zaimplementuj dwie metody pomocnicze:

- String generujPowitanie() - losuje i zwraca jedno z pięciu możliwych przywitań,
- String generujPozegnanie() - zwraca tekst składający się losowo z 1-5 słów “pa” zakończonych “!” np. “papapa!”

Wywołaj te metody i wyświetl zwrócone wartości w konsoli.

Etap 2:
Nad metodami dodaj statyczne pole przechowujące dostępne opcje programu jako tablica typu String. Przykładowe opcje: “Zagrajmy w lotto”, “wyświetl prostokąt”, “policz do 10”.
Zaimplementuj metodę void uruchomMenu() i wywołaj ją pomiędzy przywitaniem, a pożegnaniem. Metoda powinna korzystać z 3 mniejszych metod:

- void drukujListeOpcji( ) - metoda drukuje opcję z jej numerem wykorzystując wcześniej stworzone pole,
- int pobierzWybor( ) - metoda prosi o podanie opcji interesującej użytkownika tak długo, dopóki nie poda on liczby z właściwego zakresu i zwraca wybór,
- void uruchomOpcje(int) - metoda przyjmuje wcześniej pobraną liczbę i wyświetla opcję, która została wybrana.

Etap 3:
W trzech wcześniej przeniesionych klasach zmień metodę main na własną metodę. Zdekomponuj ją na mniejsze metody jeśli ma więcej niż 30 linijek kodu. W klasie Asystent zmodyfikuj metodę uruchomOpcje() tak, aby zamiast wyświetlać jaką opcję uruchomi, faktycznie to zrobiła. Na koniec zmodyfikuj, tak żeby menu powtarzało się dopóki użytkownik nie wybierze opcji “Zakończ program”.

Podpowiedź: wywołanie statycznej metody z innej klasy robimy tak, jak korzystaliśmy ze statycznych bibliotek np. Lotto.zagraj();

Projekt Bankomat

Będąc w mieście skorzystaj z bankomatu do wypłaty gotówki i przyjrzyj się dokładnie jakie opcje oferuje i w jakiej kolejności. Zadanie polega na jak najwierniejszym odtworzeniu w aplikacji działania prawdziwego bankomatu. Jeśli bankomat pyta czy chcesz potwierdzenie czy nie, Twój program też powinien pytać. Jeśli proponuje przygotowane już kwoty, Twój też powinien itd.

- **wariant trudniejszy:** Program powinien wypłacać pieniądze wyświetlając jakie banknoty i ile sztuk wydaje. Zadbaj o to, żeby bankomat wypłacał w pierwszej kolejności największymi banknotami tak, aby nie skończyły mu się zbyt prędko. Musi

więc posiadać informacje ile sztuk ma aktualnie każdego z banknotów. Jeśli kwota nie może zostać zrealizowana wydrukuj stosowny komunikat.

Projekt Gra w wisielca

Wykonaj samodzielnie implementację w konsoli następującej gry:
https://wisielec.imasz.net/wisielec.php?relaks=1&pomoc=0&kat=1

- **wariant trudniejszy:** dodaj do programu możliwość wyboru kategorii pytań oraz stopnia trudności

- **wariant trudniejszy:** to co wcześniej oraz:
    - jeśli na obu kostkach wypadnie to samo - zyskujesz 2x tyle punktów np. 2 i 2 daje 8.
- Dodaj do programu możliwość kupowania ulepszeń.
    - Za 30 punktów można wykupić dodatkową kostkę
    - za 20 punktów można zwiększyć przedział losowania o 1 na wszystkich kostkach (np z 1-6 na 1-7).

Projekt Mastermind

Wykonaj samodzielnie implementację w konsoli następującej gry:

https://webgamesonline.com/mastermind/

Programowanie obiektowe

Cechy i zachowania

43. Stwórz klasę typu Samochod oraz SamochodDemo z metodą main.
    W metodzie main stwórz dwa obiekty typu samochód.
    Obiekty typu Samochod mają mieć następujące cechy: marka, przebieg oraz przebieg do przeglądu, ta ostatnia uzupełniona jakąś wartością np 20 000 km.
    W metodzie main nadaj wartości dla cech stworzonych wcześniej samochodów.

Dla obiektów typu Samochod przygotuj następujące zachowania oraz przetestuj je w main:

- Stwórz metodę wyświetlającą markę samochodu oraz przebieg
- Stwórz metodę przyjmującą odległość do przejechania i zwiększającą przebieg samochodu o tę odległość
- Stwórz metodę zwracającą ilość kilometrów po których trzeba będzie wykonać przegląd (uwzględniając aktualny przebieg)

- **wariant trudniejszy:** stwórz w main tablicę samochodów a następnie wykorzystując pętlę wyświetl dane każdego z nich

44. Stwórz klasę Linia. Obiekty tej klasy powinny mieć dwie cechy: długość i wypełnienie. W metodzie main stwórz 3 obiekty linni i nadaj im różne długości i symbole wypełnienia. Następnie stwórz metodę void drukujLinie(), która ma drukować symbol ustawiony jako wypełnienie tyle razy, jaka jest ustawiona długość linii. np: przy ustawionej długości 6 i wypełnieniu "@" wydrukuje następującą linię:

    @@@@@@

    Przy długości 20 i wypełnieniu "*" powinna wydrukować linię:

    ****

- **wariant trudniejszy:** stwórz klasę Prostokat, i nadaj mu cechy: wysokość, szerokość oraz wypełnienie. W main przygotuj 2 obiekty tego typu i zdefinuj (zainicjalizuj) wartości w ich polach. Stwórz analogiczną metodę jak dla linni, która będzie drukować prostokąt wykorzystując informacje zawarte w jego cechach.

Konstruktory oraz metoda toString

45. W każdej z dotychczasowych klas obiektowych nadpisz i przetestuj metodę toString, która będzie zwracać w formie tekstu informacje o cechach obiektu.

46. Wróć do poprzednich zadań i przygotuj w nich konstruktory:
    - W zadaniu z liniami przygotuj konstruktor, który będzie pozwalał na ustalenie wypełnienie oraz długości linii przy tworzeniu obiektu w mainie.
    - W zadaniu z samochodami przygotuj dwa konstruktory. Pierwszy pozwalający stworzyć domyślny samochód bez przekazywania żadnych parametrów np. “Fiat 126p przebieg 300 000 tys”. Drugi konstruktor pozwalający ustawić dowolną markę i przebieg.

Obiekty pomocnicze

47. W tym zadaniu wykorzystaj wcześniej przygotowaną klasę Samochod. zaprogramuj aplikację pytającą o model Twojego samochodu oraz przebieg. Możesz zrobić to w metodzie main. Aplikacja ta ma stworzyć obiekt samochód i sprawdzić czy dany samochód powinien zrobić przegląd czy nie. Jeśli tak, powinna wyświetlić “samochód o marce … powinien zrobić przegląd już … kilometrów temu!”, a jeśli wszystko jest okej może wyświetlić “Dla samochodu o marce … zostało jeszcze … kilometrów do przeglądu”.

48. Przenieś kod obsługujący kontrolę przeglądów do osobnego obiektu typu SamochodSerwis. Powinieneś w nowej klasie uzyskać dwie metody, jedna tworząca samochód i zwracająca go a druga przyjmująca samochód i wyświetlająca dane na temat przeglądu.

- **wariant trudniejszy:** Przygotuj w main tablicę samochodów. Oblicz ich łączny przebieg wykorzystując pętlę foreach. Przenieś nową funkcjonalność do klasy CarService jako metodę przyjmującą tablicę i zwracającą int.

49. We wszystkich klasach obiektów (Samochod, Linia, Prostokat itd.) ustaw wszystkie cechy na prywatne. Jeśli wykorzystywałeś ich do przypisania startowych wartości obiektów - zastosuj konstruktor. Jeśli używałeś ich do wyświetlenie wszystkich informacji o stanie obiektu zastosuj metodę toString jeśli sprawdzałeś jedną z cech obiektu w innej klasie - zastosuj getter, jeśli zmieniałeś z innej klasy cechę obiektu zastosuj setter.

50. Zadanie powtórkowe:

Pomyśl o jakiejś rzeczy. Jeśli nie masz pomysłu wybierz coś z następującej listy:

paczka, kwiat, pies, pralka, książka, telewizor, papuga, statek, szkola, pogoda, produkt, lek, figurka, wykres, film, portfel, posiłek, państwo, buty, telefon, bankomat, zdrapka

- Przygotuj klasę do tworzenia obiektów wybranego typu oraz nadaj im 3 cechy (prywatne).
- Dla każdej z cech zastanów się jak ma być uzupełniana, zawsze przy tworzeniu obiektu jak np. cena produktu czy automatycznie ustawiana na stałą wartość (przebieg = 0 dla nowego samochodu) a może ma być losowana lub generowana? Przygotuj odpowiedni/e konstruktory.
- Stwórz 2 różne przykładowe obiekty w mainie. Wyświetl je wykorzystując metodę toString
- Przygotuj dwa zachowania (metody), zastosuj w conajmniej jednej z nich zwracanie lub przyjmowanie wartości np.
    - Kwiat: zakwitnij()
    - Papuga: uczSieSlowa(String slowo)
    - Figurka: pomaluj(String kolor)
- przygotuj pomocniczą klasę która wykorzysta stworzony obiekt i wykona jakieś zadanie z nim związane np:
    - Ogród: zerwijZakwitnieteKwiaty(Kwiat[] kwiaty)
    - Pirat: nakarmPapuge(Papuga papuga)
    - Kolekcjoner: pomalujCzarneFigurki(Figurka[] figurki)
- pamiętaj, żeby stosować gettery, settery oraz własne metody aby posługiwać się swoimi obiektami np:

```
public void podpiszKsiazke(Ksiazka ksiazka){
     if ( ! ksiazka.isPodpisana() ) {
          ksiazka.setPodpis("J. K. Rowling");
     }
```

```
}
```

Poczta part 1

Zaprojektuj i przetestuj nowy, referencyjny typ danych: **Paczka**.

- Paczki mogą mieć **nadawcę**, **odbiorcę** oraz **wagę**, wyrażoną z dokładnością do gramów.
- Mogą również być **priorytetowe** lub nie, domyślnie nie są.
- Tworząc paczkę trzeba podać nadawcę, odbiorcę oraz jej wagę.

Konstruktor powinien skontrolować te wartości i wyświetlić błąd jeśli:

- adresat lub nadawca nie został podany
- waga nie mieści się w zakresie 1- 1000.

skontroluj, czy cechy paczek ustawiają się poprawnie

Przygotuj **drugi** alternatywny sposób tworzenia paczek (👉konstruktor), który będzie **losował** imię nadawcy, imię odbiorcy, wagę paczki i to czy jest ona priorytetowa czy nie.

Przygotuj również **metodę** pozwalającą wyliczyć i zwrócić **cenę** paczki, która zostanie wyliczona na podstawie jej parametrów:

- paczka do 0.5 kg kosztuje 5 zł
- do 1kg kosztuje 8zł
- do 2 kg kosztuje 12 zł
- powyżej 2kg doliczane jest 1 zł za każdy kilogram
- paczka kosztuje 10% drożej jeśli jest priorytetowa

manipulacje zmiennymi:

- Przygotuj w main **dwie zmienne** z paczkami
- **Stwórz jedną zmienną bez paczki**, zainicjowaną wartością **null**.
- Sprawdź **cenę** paczek wykorzystując te 3 zmienne.
- Napisz instrukcję, która wstawi pierwszą paczkę do trzeciej zmiennej i zastąpi nulla
- wyświetl cenę paczki w trzeciej zmiennej
- Przetestuj co się stanie jeśli ustawisz wszystkie zmienne jako “**final**”.

Przenoszenie obiektów

Poczta part 2

Tablice paczek

- Przygotuj w main **tablicę 3 paczek**

- uzupełnij ją paczkami
- wyświetl dane pierwszej paczki z tablicy.

- Stwórz **tablicę 100 paczek**
- uzupełnij ją losowymi paczkami
- .Osobną pętlą wyświetl wszystkie wylosowane paczki z tablicy (👉for each)
- Wyświetl również łączną cenę wysłania tych paczek.

Rozbudowa programu

- Dodaj do typu Paczka nową cechę **status**.
- domyślnie ma być on ustawiony na “**utworzona**”
- Przygotuj nowy typ danych: **Poczta**.

zachowania poczty

- Do poczty dodaj metodę, która pozwoli nam **nadawać paczki**
- Metoda ma wymagać przekazania paczki oraz kwoty pieniędzy
- Metoda przyjmując paczkę powinna sprawdzić jej cenę, i jeśli zapłacono wystarczającą ilość pieniędzy ma zmienić jej status na “nadana” oraz zwrócić resztę.
- Jeśli przekazano zbyt małą kwotę ma zwrócić tę kwotę i nie modyfikować stanu paczki.
- Na koniec dodaj do poczty pole “utarg” i dodawaj do niego zarobione przez pocztę pieniądze.

- Dodaj metodę pozwalającą na **stworzenie i zwrócenie paczki**
- Ma ona zapytać **w konsoli** o wszystkie parametry paczki i stworzyć ją jako obiekt
- na koniec zwróć utworzoną paczkę

- W mainie spróbuj wykorzystać nową metodę do wytworzenia paczki
- nadaj ją przy pomocy poprzedniej metody do nadawania paczek

- **Dodatkowe:** Dodaj metodę, która będzie wyświetlała ostatnią nadaną paczkę.

Adres

- Przygotuj nowy typ danych **Adres**.
- Adres powinien zawierać cechy takie jak: **ulica**, **numer domu** oraz **kod pocztowy**.
- Tworząc obiekt adres należy przekazać wszystkie dane przez konstruktor.

List

- Przygotuj typ **List**.
- List powinien zawierać **adres nadawcy**, **adres odbiorcy** oraz może być **priorytetowy** lub nie.
- Podobnie jak paczka powinien zawierać pole **status** które domyślnie ustawione jest na “**nienadany**”.

- Aby stworzyć list należy **przekazać** adres nadawcy, i adres odbiorcy (jako obiekt typu **Adres**).
- Przetestuj obiekt typu Adres tworząc List i wyświetlając go w main.

- Dodaj **metodę** zwracającą **cenę** listu.
- Jeśli list jest priorytetowy, kosztuje 8,50zł jeśli nie, 6 zł.

Wysyłanie listów

- Od teraz każda **poczta** ma posiadać **pustą tablicę listów** (**pole**), do której jest w stanie zmieścić **10 listów**.

- Przygotuj **metodę nadawania listów**
- która będzie działała podobnie do tej nadającej paczki z tym, że jeśli list został poprawnie opłacony powinien trafić na **pierwsze wolne miejsce w tablicy**
- Jeśli tablica jest **pełna** metoda powinna wyświetlić, “przepraszamy, poczta jest w stanie wysyłać tylko 10 listów dziennie, i co nam zrobisz?”, chyba, że masz pomysł jak inaczej rozwiązać ten problem :)

- Dodaj do poczty możliwość **wysłania listonosza**
- **Metoda** ta ma ustawić status wszystkich listów na “**wysłany**”.

Projekt Dom

51. W tym zadaniu trzeba będzie stworzyć kilka obiektów, nadać im odpowiednie **cechy**(pola), przygotować **konstruktory** narzucające odpowiedni sposób uzupełniania tych cech oraz po jednym zachowaniu (metoda). Po stworzeniu każdego z obiektów przygotuj dla niego metodę **toString** i przetestuj w main czy budowanie go oraz metoda działa poprawnie tzn. cechy obiektu mają odpowiednie wartości.

    Zrób osobny **pakiet**, w którym zrealizujesz następujące założenia
    - Obiekty typu **Okno** mogą być otwarte lub zamknięte
    - Nowo stworzone okno jest **zamknięte** ale można je później otworzyć
    - Przygotuj w programie **tablicę 4 okien** i otwórz wszystkie okna
    - Obiekty typu **Lozko** posiadają licznik, ile dni minęło od ostatniej wymiany pościeli
    - Przy każdym tworzeniu łóżka trzeba przekazać ile dni temu miało przebieraną pościel (nie używaj Scannera tylko parametrów konstruktora)
    - Obiekty typu **Pokoj** posiadają jedno łóżko oraz tablicę okien
    - Są dwa sposoby stworzenia pokoju. Przekazując łóżko oraz tablice okien oraz drugi wykorzystujący bezparametrowy konstruktor
    - Tworząc pokój drugim sposobem w pokoju ma stworzyć się tablica z 2 oknami oraz łóżko nie przebierane od 5 dni

- Pokój można **posprzątać** wywołując odpowiednią metodę. Ma ona przebrać pościel oraz pootwierać wszystkie okna
- Obiekty typu **Dom** posiadają tablicę pokojów
- Aby stworzyć dom należy przekazać pokoje przez konstruktor
- W programie stwórz dom z dwoma pokojami: jeden domyślny, a drugi, większy z czterema oknami

Projekt ogród

//Przygotowanie obiektow i powiazan
Przygotuj klasy obiektowe, nadaj obiektom cechy oraz zadbaj o ich uzupełnianie wg następujących wytycznych:
Lawka moze byc z określonego materiału(drewno, stal itp)
rodzaj materiału należy przekazać przez konstruktor przy tworzeniu obiektu
Drzewo ma określoną ilość owoców. Wartość ta losuje się od razu z zakresu od 100 do 1000
Pies ma imie i zna tablice sztuczek
Aby stworzyć psa należy przekazać mu imię i jakie zna sztuczki (np: aport, daj lape...)
Kwiat kwitnie lub nie i ma jakis kolor (String lub enum)
Nowo stworzony kwiat nie kwitnie.
Kwiat można stworzyć na dwa sposoby, można podać kolor lub nie.
Jeśli nie podamy koloru to powinien wstawić się losowy kolor z 4 możliwych.
(zastosuj dwa konstruktory)

//zbiorczy obiekt Ogrod
Ogrod moze miec drzewa, lawke, psa i kwiaty.
Aby stworzyć ogród należy:
przakazać obiekt ławki
przekazać tablicę kwiatów
przekazać ile ma być drzew - tablica drzew odpowiedniej wielkości ma się stworzyć automatycznie oraz wypełnić obiektami drzew
przekazać psa lub nie (możesz zastosować dwa różne kontruktory)

//MetodyObiektow - Dzialania obiektow
Kwiat: Jesli wywolamy metode podlejKwiat() kwiat kwitnie
Drzewo: Jesli wywolamy metode owocuj() drzewu przybywa od 1 do 100 owocow
Drzewo: Jesli wywolamy metode zbierajOwoce() zwroci aktualna ilosc owocow i wyzeruje owoce
Pies: Jesli wywolamy metode - bawSie() wyświetli jaką pies o danym imieniu robi sztuczkę (wylosuj)

//MetodyOgrodu
Jesli wywolamy metode wejdzDoOgrodu() przywita nas pies i zrobi sztuczke lub przywita nasszum wiatru jesli psa nie ma
Jesli wywolamy metode podlejKwiaty() wszystkie kwiaty maja byc podlane
Jesli wywolamy metode odpocznij() ma zostac wyswietlone "siadasz na lawce (z materiału: ...)".

Jesli wywolamy metode zbierajOwoce() metoda powinna zebrac owoce ze wszystkich drzew i wyświetlić ile ich było.
Jesli nie bylo zadnych owocow metoda powinna wyswietlic "przyjdz jutro!"
Jesli wywolamy metode wyjdzZogrodu() drzewa maja owocowac a kwiaty przestaja kwitnac (może potrzebna dodatkowa metoda w kwiecie?)

Jesli wywolamy metode pracujWogrodzie() metoda ta wywola pokoleji wszystkie metody ogrodu

//Dla ambitnych
Kazda praca w ogrodzie to jeden dzien. Ulepsz program aby mozna bylo pracowac w ogrodzie przez miesiac i
aby program wyliczył ile wyprodukowal owocow

Projekt 7: Warsztat samochodowy

Przygotuj program w którym zasymulujesz sytuację w której kierowca samochodu **łapie gumę w kole i zabiera go do warsztatu** do naprawy.

- w main przygotuj **4 obiekty kół**, każde z nich ma mieć ustawione **ciśnienie** i ma **nie być przebite**
- zbierz wszystkie koła do **zbioru** i **przekaż** do obiektu typu **Samochod**
- zasymuluj zdarzenie, że **samochód łapie gumę w losowym kole**
- obiekt typu **Warsztat** może **przyjmować samochód** do **naprawy**
- naprawa polega na tym, że warsztat wyszukuje które koło z samochodu jest zepsute i poprawia jego stan
- na koniec naprawy warsztat zwraca obiekt typu **Paragon** zawierający **rodzaj usługi**, **ile razy była wykonana** oraz **łączną cenę**

52. Fabryka Linii

W programie z Liniami stwórz dodatkową klasę: FabrykaLinii, a w niej stwórz metody, wytwarzające i zwracające:

a) linię losowej długości z wybranym znakiem,
b) linię wybranej długości ale z losowym znakiem,
c) linię z losową długością i losowym znakiem
d) określoną ilość linni z losową długością i znakiem (wykorzystaj wcześniejszą metodę)
e) linię z długością 2 i losowym znakiem ale każda kolejna zwrócona linia ma być 2x większa od poprzedniej

Następnie w klasie LinieDemo przetestuj wszystkie metody drukując wyprodukowane przez nie linie.

Listy

53. Stwórz listę imion i uzupełnij je 5 imionami.
    - Wyświetl wszystkie imiona
    - Wyświetl pierwsze imię
    - Zmień drugie imię na “Marcin”
    - Usuń jedno z imion
    - Stwórz drugą listę z dwoma kolejnymi imionami i dodaj do niej wszystkie imiona z poprzedniej za pomocą metody addAll
    - Wyświetl wszystkie imiona z drugiej listy wielkimi literami
    - Wyświetl tylko żeńskie imiona (umownie - te kończące się na ‘a’)

54. Stwórz klasę abstrakcyjną Citizen oraz klasy dziedziczące:
    - Peasant(Chłop),
    - Townsman(Mieszczanin),
    - King(Król),
    - Soldier(Żołnierz)

    Wszystkie klasy posiadają pole imie (przemyśl gdzie powinno się znajdować to pole). Citizen powinien być klasą abstrakcyjną która posiada metodę abstrakcyjną 'canVote' która zwraca true dla townsman'a i soldier'a, ale false dla chłopa i króla.

    Stwórz klasę Town która posiada zbiór obiektów typu Citizen. Dodaj do niej kilku citizenów (tworząc ich w main). Obiekt klasy Town ma mieć metody:
    - howManyCanVote - które zwracają ilość osób które mogą głosować
    - whoCanVote - która wypisuje imiona osób które mogą głosować