Задача 3. (15 т.)
Да се напише (на език по избор - Java или C++) клас **преносим компютър**, с параметри - включен към безжична мрежа (от булев тип), размер на паметта в гигабайти, и поле от подходящ тип, което да съхранява честотата на процесора в гигахерци (например 2,4 Гигахерца). В класа да има 3 конструктора - един без параметри, един за копиране и един за общо ползване (със всички параметри). Класът да позволява да се запомня броят на включените към безжична мрежа компютри. Да се напишат методи за достъп (set/get) методи за полето за брой на процесорите и за полето за състояние (включено/изключено).

Задача 4. (15 т.)Да се напише програма (алгоритъм) - на програмен език по избор (C++ или Java), която намира броят на еднаквите последователни елементи в свързан списък.

Задача 3. (15 т.)
Да се напише (на език по избор - Java или C++) клас **компютър**, с параметри - брой процесори, размер на паметта в гигабайтове и поле от булев тип, което задава дали компютърът е включен/изключен. В класа да има 3 конструктора - един без параметри, един за копиране и един за общо ползване (с всички параметри). Класът да позволява да се запомня броят на създадените компютри. Да се напишат методи за достъп (set/get) методи за полето за брой на процесорите и за полето за състояние (включено/изключено)

Задача 4. (15 т.) Да се напише програма (алгоритъм) - на програмен език по избор (C++ или Java), която:

a) реализира клас **стек** от знаци;
b) проверява дали в даден низ затварящите кръгли скоби съответстват на отварящите кръгли скоби (чрез използване на стек).

Задача 3. (10 т.) *Задачата да се реши на езика C, C++ или Java. В началото на вашето решение посочете кой език сте избрали*.

Да се напише програма, която създава **едномерен масив** A с 27 елемента и след това:

a) Генерира по случаен начин стойности в интервала (0, 4.9) за елементите на така зададения масив A;
b) Разпечатва данните от масива в 5 КОЛОНИ, по диапазони, както следва: [0,1), [1,2), [2,3), [3,4), [4,5) Етикетите на колоните да бъдат съответно „Под 1", „Под 2", „Под 3" и т.н. – както е показано в примера по-долу. При извеждането данните да бъдат подравнени и форматирани по следния начин: всяка колона да бъде с широчина 8 символа, числата в колоните да бъдат с 4 знака след запетайката и изведени с дясно подравняване, а етикетите – с ляво подравняване.

Примерен резултат от програмата*:

```
Под 1    Под  2   Под 3    Под 4    Под 5
   0,004    1,028    2,068    3,272    4,376
   0,156    1,412    2,656    3,724    4,688
    0,22    1,432     2,96    3,828    4,716
   0,272    1,436                     4,748
   0,288    1,544                      4,82
   0,396    1,768                     4,888
   0,416    1,992
    0,46
```

* Забележка: НЕ Е необходимо резултатите във всяка от колоните да бъдат сортирани.

Задача 4. (10 т.) *Задачата да се реши на езика C++ или Java. В началото на вашето решение посочете кой език сте избрали*.

А) Да се дефинира структура `ChessPosition`, описваща коректна позиция на фигура върху шахматна дъска (координатите на позицията са от ‘A’ до ‘H’ по едното измерение и от 1 до 8 по другото).

Да се дефинира абстрактен клас (или интерфейс) `ChessPiece`, описващ шахматна фигура със следните операции:

- `ChessPosition getPosition()` – Връща позицията на фигурата на дъската;
- `[подходящ тип] allowedMoves()` – Връща списък (колекция) с всички възможни позиции, до които дадена фигура може да достигне с един ход;
- `[булев тип] captures(ChessPosition pos)` – Проверява дали фигурата “владее” позицията `pos`, подадена като параметър, т.е. дали позицията е в списъка с възможните ходове на фигурата. Булевият тип да бъде булевият тип в езика, който сте избрали (напр. `bool`, ако пишете на C++).

Б) Да се дефинират класовете `Rook` и `Knight` – наследници на `ChessPiece`, описващи съответно шахматните фигури топ и кон.

В) „Стабилна конфигурация“ наричаме такава подредба на фигурите върху дъската, при която никоя фигура да не е върху позволен ход на друга фигура (т.е. никои две фигури да не се „бият“). Да се дефинира функцията `allMoves ([подходящ тип] pieces[, …])`, която за списъка (колекцията) `pieces`, съдържащ произволен брой разнородни шахматни фигури, отпечатва на конзолата всеки възможен ход на фигура от `pieces` такъв, че след изпълнението му списъкът с фигури да описва стабилна конфигурация. Информацията за ходовете да съдържа типа на фигурата, старата позиция и новата позиция, например:

```
Rook A1 -> B1
Knight B3 -> A5
```

Забележка: Реализирайте всички конструктори и други операции, които смятате, че са необходими на съответните класове.

Задача 3. (10 т.) *Задачата да се реши на езика C++ или Java. В началото на вашето решение посочете кой език сте избрали*.

Дадени са координатите на N-точки, които са записани в масивите `float x[N], y[N]` по следния начин: координатите на `i`-тата точка са `(x[i], y[i])`.

Напишете функция `square`, която получава като аргументи броя на точките N и два масива `X` и `Y` съдържащи координатите им и извежда на екрана координатите на центъра и страната на най-малкия квадрат със страни успоредни на координатните оси, който обхваща всички дадени точки (всички дадени точки са във вътрешността му или на страните му).

Ако решавате задачата на Java, достатъчно е да напишете статична функция, която решава задачата.

Задача 4. (10 т.) *Задачата да се реши на езика C++ или Java. В началото на вашето решение посочете кой език сте избрали.*

Нека GameBoard е предварително дефинирана квадратна матрица от цели числа с размери N x N, представяща игрова дъска. Всеки елемент в матрицата има стойност 0 („земя"), 1 („огън") или 2 („вода"). За две позиции в матрицата (i, j) и (i', j') казваме, че са съседни, ако $|i - i'| \leq 1$ и $|j - j'| \leq 1$.

А) Да се дефинира структура `Point`, описваща позиция на игровата дъска. Да се дефинира абстрактен клас (или интерфейс) `GamePlayer`, който описва играч на игровата дъска със следните операции:

- `getPosition`() – Връща позицията на играча на дъската;
- `allowedMoves`() – Връща списък (колекция) с всички възможни позиции, до които играчът може да достигне с един ход.

Б-1) Да се дефинира клас `Knight`, наследник на `GamePlayer`, описващ „сухопътен рицар". Рицарят може да се придвижва само в такава съседна позиция, която е „земя" и не е в съседство с „огън". Пример за достижими позиции за рицаря `K` е показан на диаграмата вдясно.

1	0	1
0	K	2
0	**0**	2

Б-2) Да се дефинира клас `SeaMonster`, наследник на `GamePlayer`, описващ „морско чудовище". Морското чудовище може да се придвижва с произволен брой позиции по хоризонтала или по вертикала, но само по „вода". Пример за достижими позиции за чудовището `S` е показан на диаграмата вдясно.

1	1	0	2	0
0	**2**	1	0	2
2	S	**2**	**2**	1
1	1	2	2	0
2	2	1	1	1

В) „Война" наричаме такава подредба на играчите по дъската, при която на някоя от съседните позиции на всеки играч има друг играч. Да се дефинира функция

```
allMoves ([подходящ тип] players[, …])
```

която по даден списък (колекция) `players`, съдържащ произволен брой разнородни играчи, извежда на стандартния изход всеки възможен ход на играч от `players` такъв, че след изпълнението му списъкът с играчи да описва война. Информацията за ходовете да съдържа типа на играча, старата позиция и новата позиция.

Пример:

```
Knight (0,0) -> (1,1)
SeaMonster (2,2) -> (5,2)
```

Забележка: реализирайте всички конструктори и други операции, които смятате, че са необходими на съответните класове.