Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

ЗАДАЧИ ЗА ДОМАШНО 1

курс Увод в програмирането
за специалности Информатика, Компютърни науки и Софтуерно инженерство
зимен семестър 2016/2017 г.

ВАЖНО: В решенията на дадените по-долу задачи **не трябва** да се използват цикли, нито операторът `goto`. Всяка задача носи по две точки.

ВАЖНО: всички задачи ще бъдат тествани автоматично, затова е важно извеждането на екрана да спазва абсолютно точно това, което е указано в задачата! Едно от изискванията е всеки изведен ред да завършва със знак за нов ред (`'\n'`). На лекции ще получите повече информация.

ВАЖНО: всички задачи ще бъдат проверени автоматично за преписване. Файловете с голямо съвпадение ще бъдат проверени ръчно от лектора и при установено заимстване ще бъдат анулирани.

Задача 1. Да се напише програма, която прочита от клавиатурата цяло положително число, не по-голямо от 3000, и извежда на екрана представянето му в римски цифри. Ако числото е извън границите, да се изведе текстът `"Invalid number!"`. Например:

Вход	**Изход**
`1990`	`MCMXC`
`5000`	`Invalid number!`

Задача 2. Да се напише програма, която прочита цяло положително число с 10 цифри — ЕГН на български гражданин. Програмата трябва да провери дали даденият ЕГН е валиден и ако да — да изведе на екрана разделени с интервал датата на раждане (във

формат ДД.ММ.ГГГГ) и пола на съответния човек (главна латинска буква M или F). Ако въведените данни не са валидно ЕГН, да се изведе текста `"Bad input data!"`.
Удобно обяснен форматът на ЕГН може да намерите тук и тук. Например:

Вход	**Изход**
`9801010800`	`01.01.1998 M`
`7816031663`	`Bad input data!`

Задача 3. Да се напише програма, която прочита цяло число без знак (по-малко от 2^{32}) и проверява дали шестнайсетичният му запис е съставен от еднакви цифри, като в зависимост от резултата извежда съответно текста `"Yes"` или `"No"`. Например:

Вход	**Изход**
`1365`	`Yes`
`100`	`No`

Задача 4. Да се напише програма, която по дадени координати на цар и друга фигура върху шахматна дъска определя дали царят е в шах от съответната фигура.

Координатите на шахматната дъска са във вида - “X Y”, където:

- X е малка латинска буква от ‘a’ до ‘h’, означаваща поредна колона на шахматната дъска, отляво надясно
- Y е цифра от 1 до 8, означаващо пореден ред на шахматната дъска, отдолу нагоре

Фигурата може да бъде: дама, офицер, кон или топ. Символите, съответстващи на валидните фигури са:

- ‘Q’ — дама (Queen)
- ‘B’ — офицер (Bishop)
- ‘N’ — кон (kNight)
- ‘R’ — топ (Rook)

(Удобно описание на шахматната нотация можете да намерите тук.)

От клавиатурата последователно се въвеждат:

1. Символ за тип на атакуващата фигура
2. Координатите на атакуващата фигура
3. Координатите на царя

Ако царят е в шах, програмата извежда надпис "Yes" на конзолния ред, а в противен случай извежда "No".

Пример:

Вход	**Изход**
R c 5 g 5	Yes
B h 3 d 2	No
N c 3 d 5	Yes

Задача 5. След поредна нощ из столичните нощни заведения, вицесветовният шампион по дартс и първокурсник във ФМИ, Интегралчо бива предизвикан да покаже своите умения със стреличките от Вальо Точния, виден квартален шмекер, носещ гордо прозвището "батка". Без дори да се замисли, Интегралчо приел и се запътили към най-близката отворена игрална зала. Изправяйки се пред игралното табло, вицесветовният шампион установил, че има проблем. Осъзнал, че след приетото количество течни субстанции картината пред него е "леко" разфокусирана. Въпреки това, той не загубил самообладание, знаейки, че всеки негов изстрел попада точно на избраните от него координати. На Точното не може да му се има доверие, а Интегралчо не може да разчита на своето зрение, затова няма как да разбере колко точки е изкарал. В замъгленото съзнание на студента от ФМИ се ражда проста, но гениална мисъл — той се сеща за вашите способности да пишете код на C++.

За да помогнете на изпадналия в беда ваш колега, вие ще трябва да реализирате програма, която от стандартния вход прочита две двойки координати *(x,y)* и *(u,v)* (четири дробни числа, разделени с интервал). *(x,y)* са координатите на точката, в която Интегралчо се цели, а *(u,v)* показва отклонението на погледа му. На стандартния изход вие трябва да кажете този изстрел колко точки ще му донесе. Ще приемаме, че дъската за дартс е разделена на области от 3 концентрични окръжности, с радиуси $R_1 > R_2 > R_3$ (предварително известни константи). Центърът му при "трезвен" поглед съвпада с началото на координатната система ($u = v = 0$). Пречупено обаче през погледа на вашия приятел, центърът на координатната система отговаря на *(u,v)*, което дефинира нова координатна система с начало тази точка. Интегралчо се цели спрямо това, което вижда

(новата координатна система). Помогнете му да разбере колко точки получава в действителност за неговия изстрел.

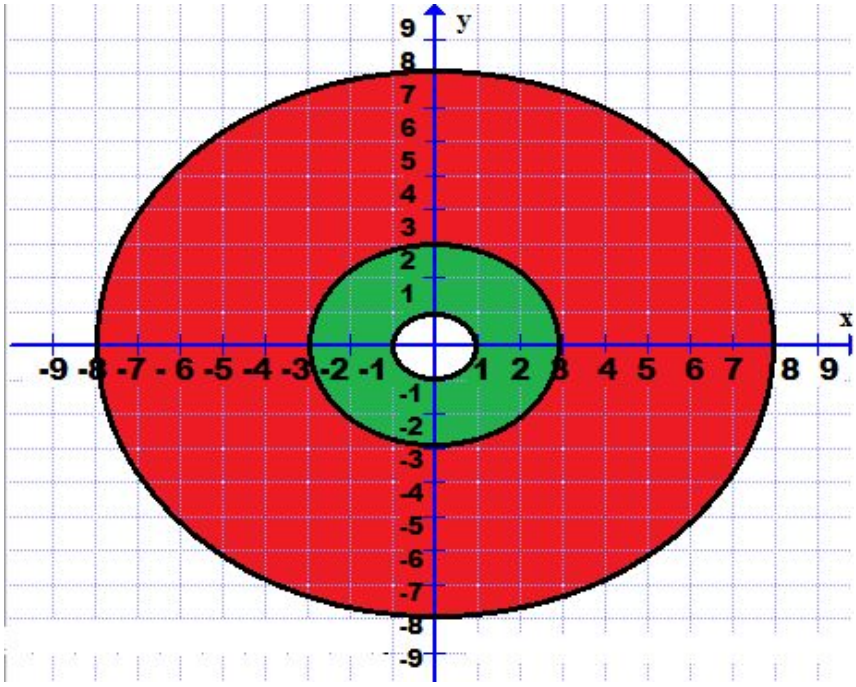
Точкообразуване:
Ако Интегралчо уцели извън окръжностите на дъската за дартс, той получава нула точки. В случая, когато стреличката му попадне в най-големият кръг, получава 10 точки. Ако точката, която уцели, принадлежи на вторият по големина кръг, точките, които ще получи, са два пъти повече. А ако вашия приятел улучи центъра (най-малкия кръг), то той получава общия брой на точките от предишните две области, умножен по две. Радиусите на различните области са съответно $R_1 = 8$, $R_2 = 3$ и $R_3 = 1$.
Редакция от 10.11.2016 г.: *за определеност, ако стреличката попадне точно на границата между два сектора, ще считаме, че се получават нула точки. (Можем да мислим, че отскача от телта между двата сектора и пада на земята)*

Помогнете на вашия колега да победи Вальо Точното като му кажете, ако се цели в дадена точка, колко точки реално ще му донесе тя.
Пример: ако входните данни са (3,3) – точката, която цели и (1,2) – отклонение на погледа му, то той ще получи десет точки.

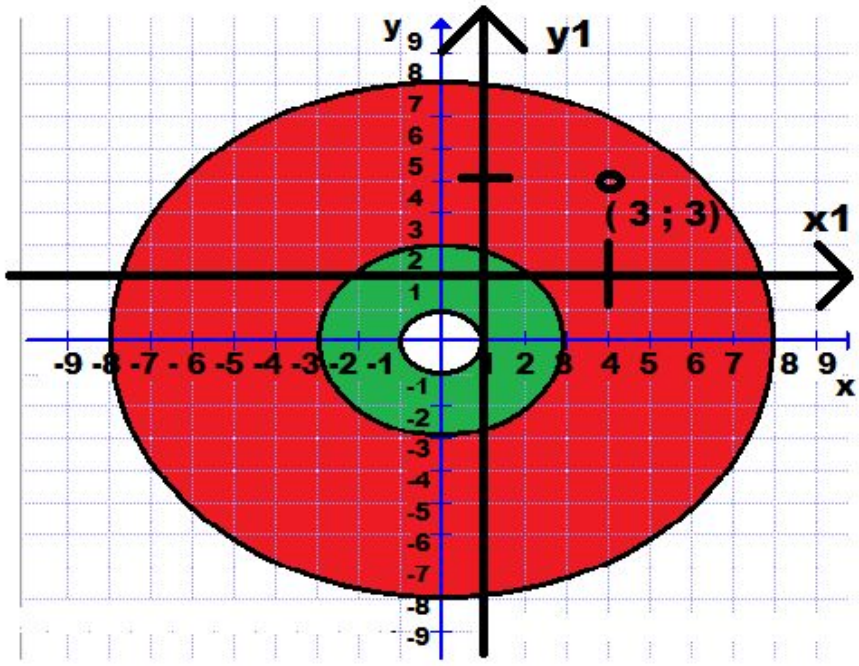
Пример:

Вход	**Изход**
3 3 1 2	10

Дартсът през “трезвен” поглед:



Дартсът през прогледа на Интегралчо :



** удебелената координатна система е тази, по която цели Интегралчо*

Други примери:

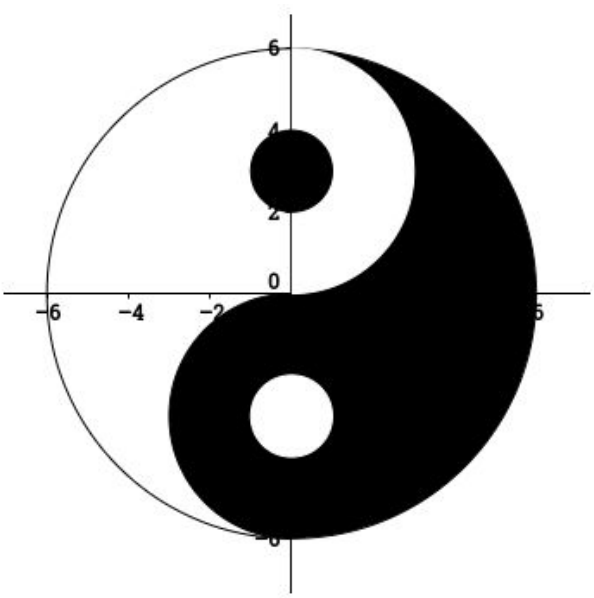
Вход	Изход
0 0 1 1	20
0 0 -0.5 -0.5	60
5 5 2 2	0

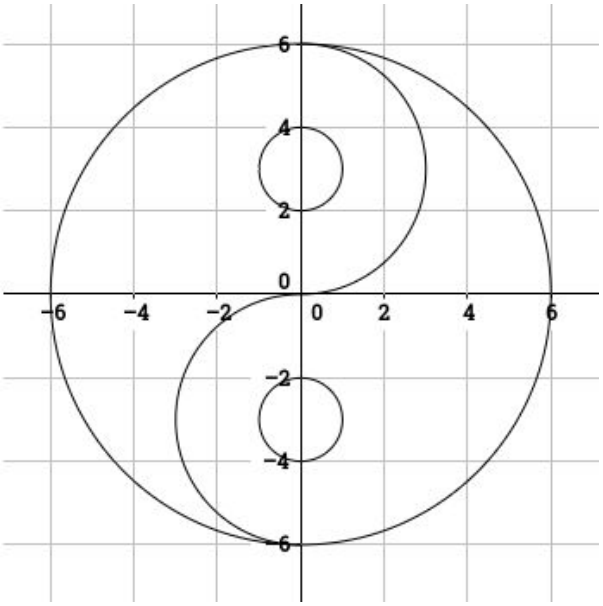
Забележка: в рамките на тази задача точността на сравненията да бъде извършвана с до 3 значещи цифри след десетичната запетая!

Задача 6. Дадена е декартова координатна система. Всеки правоъгълник със страни успоредни на осите може да се определи с четири цели числа: координатите на долния си ляв ъгъл, височината и ширината си. Да се напише програма, която прочита от клавиатурата данните за два правоъгълника (на два отделни реда, като числата на всеки ред са разделени с интервал) и извежда на екрана лицето на общата им част. Ако правоъгълниците нямат обща част, приемаме лицето за нула. Например:

Вход	Изход
1 1 5 6 -3 4 6 8	8
1 1 5 6 3 10 4 2	0

Задача 7. Дадена е долната фигура “ин-ян” в декартова координатна система. Да се напише програма, която прочита от клавиатурата координатите на точка (две дробни числа, отделени с интервал) и определя дали точката е в “злата” (черната) или в “добрата” (бялата) част на фигурата или е на границата между двете. Ако точката е извън фигурата, да се приеме за гранична. Програмата да извежда на стандартния изход съответно `"Evil"`, `"Good"` или `"Neutral"`.





Примери:

Вход	**Изход**
`1 -1.5`	`Evil`
`2.5 3`	`Good`
`0 2`	`Neutral`

Задача 8. Разполагаме с банкноти от 50, 20, 10, 5, 2 и 1 лева. Дадена сума може да бъде представена по много различни начини от банкнотите. Например 286 = 2×50 + 5×20 + 8×10 + 3×2, но и 286 = 5×50 + 1×20 + 1×10 + 1×5 + 1×1. Интересуваме се от най-икономичното представяне на дадена сума, т.е. това представяне, което изисква най-малък брой банкноти. Да се напише програма, която приема от клавиатурата сума (цяло положително число) и намира най-икономичното представяне на тази сума чрез банкнотите. Представянето да бъде изведено на стандартния изход съгласно следния синтаксис, описан чрез метаезика на Backus-Naur:

```
<сума> = <брой>*<номинал> { + <брой>*<номинал> }
```

Номиналите на банкнотите в представянето на сумата да са подредени в намаляващ ред.
Упътване: Номиналите на банкнотите са подбрани така, че получаваме най-икономичното представяне последователно подбирайки банкноти от възможно най-високия номинал, докато това е възможно, след което преминаваме към по-ниския номинал.

Примери:

Вход	**Изход**
286	286 = 5*50 + 1*20 + 1*10 + 1*5 + 1*1
88	88 = 1*50 + 1*20 + 1*10 + 1*5 + 1*2 + 1*1

Задача 9. Искаме да симулираме работата на 4-битов калкулатор, който работи само с неотрицателни цели числа от 0 до 15. За целта на стандартния вход приемаме 2 числа и символ, означаващ операция, в реда: <число> <операция> <число> (например 5 + 4). Калкулаторът трябва да пресметне резултата от операцията и да го изведе на екрана. Калкулаторът трябва да поддържа аритметичните операции за цели числа - събиране (+), изваждане (-), умножение (.), частно (/), остатък (%). Понеже е само 4-битов, при калкулатора се получава препълване (overflow), ако резулатата от операцията е прекалено голям. Препълване се получава и ако входните числа са прекалено големи. Например 5 - 20, след отчитане на препълването на 20, ще стане на 5 - 4 = 1 (понеже 20 - 16 = 4 след препълването). Също така 9 + 8 = 17, което след отчитане на препълването дава 1. На стандартния изход да бъде изведен резултата от операцията, като при деление на 0 на екрана се извежда низа `"Division by zero!"`.

Пример:

Вход	**Изход**
5 - 20	1
9 + 8	1
3 - 7	12
5 % 0	Division by zero!

Задача 10. Да се напише програма, която по зададени коефициенти на система от 2 линейни уравнения с 2 неизвестни намира решението на системата, ако то е единствено.

Вход: два реда, на всеки от които три дробни числа (с до пет значещи цифри), разделени с интервал и означаващи коефициентите пред неизвестните x и y от лявата страна на равенството и числото от дясната страна на равенството.

Изход:

- ако системата има единствено решение, две дробни числа, разделени с интервал и означаващи стойностите на x и y. Дробните числа да бъдат изведени с точност пет значещи цифри (напомняме, че “брой значещи цифри” и “брой цифри след десетичната точка” са различни неща).
- ако системата няма решение, низа `"No solution"`
- ако системата има повече от едно решение, низа `"Many solutions"`

Примери:

Вход	**Изход**
`1.2 2.3 13.87` `5.3 -0.5 17.56`	`3.7 4.1`
`1.2 2.3 13.87` `1.8 3.45 20.805`	`Many solutions`
`1.2 2.3 13.87` `3 5.75 34.5`	`No solution`