# Упражнения: Вложени условни конструкции

Задачи за упражнение в клас и за домашно към курса [„Основи на програмирането“ @ СофтУни](https://softuni.bg/courses/programming-basics).

## Празен Visual Studio Code проект

Създайте празен проект във Visual Studio Code. Ще обединяваме решенията на всички задачи, под формата на отделни файлове в този проект. Тази възможност е изключително удобна, когато искаме да работим по няколко проекта и бързо да превключваме между тях или искаме да обединим логически няколко взаимосвързани проекта. Това ни помага да запазваме решенията на задачите отделно и да ги пазим, за да ги използваме за други задачи или преговор.

1. Стартирайте Visual Studio Code
2. Създайте нова папка, която ще държи отделните решения. Ще се отвори диалогов прозорец, в който ще трябва да изберете нейната директория. Препоръчително е да именувате папката според темата на заданието, пример "**Simple-Conditions**"



След това изберете папката като работна среда, за да добавяте файловете с JavaScript решенията на своите задачи в нея.



Панелът в ляво ще изглежда така:



## Обръщение според възраст и пол

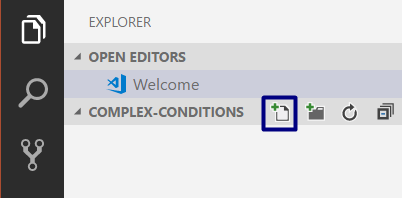
Първата задача от тази тема е да се напише **конзолна програма**, която **прочита възраст** (десетично число) и **пол** (“m” или “f”), въведени от потребителя, и отпечатва **обръщение** измежду следните:

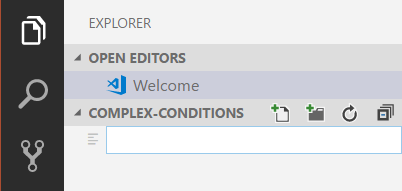
* “Mr.” – мъж (пол “m”) на 16 или повече години
* “Master” – момче (пол “m”) под 16 години
* “**Ms.**” – жена (пол “f”) на 16 или повече години
* “**Miss**” – момиче (пол “f”) под 16 години

### Примерен вход и изход:

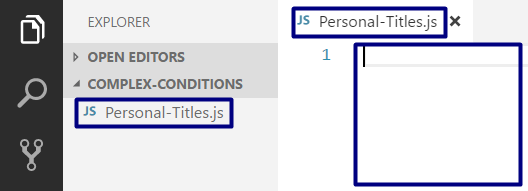
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 12  f | Miss | 17  m | Mr. | 25  f | Ms. | 13.5  m | Master |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте подходящо. Препоръчително е всеки скриптов файл да се казва както името на задачата, чието решение съдържа.





1. Съдържанието на новият файл ще се отвори в прозореца в дясно.



1. Отидете във файла Personal-Titles.js и създайте функцията personalTitles([arg1, arg2]). Можете да си помогнете с картинката по-долу:

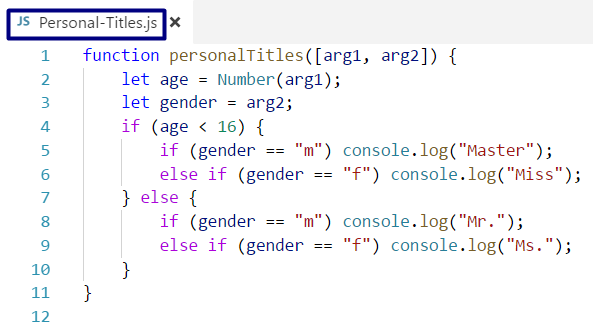
function personalTitles([arg1, arg2]) {

//Your code goes here...

}

//Call the function to execute the code inside

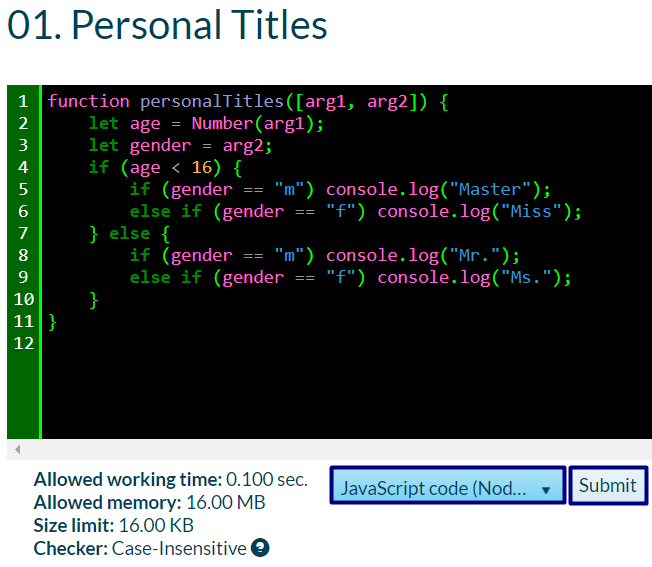
personalTitles(["25", "f"]);

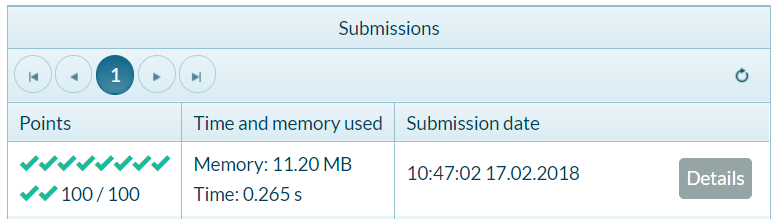


1. **Стартирайте** програмата с Ctrl + [F5] и я **тествайте** с различни входни стойности.



1. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#0>. Трябва да получите **100 точки** (напълно коректно решение):





## Квартално магазинче

Следващата задача има за цел да тренира работата с **вложени проверки** (nested if). Ето го и условието: предприемчив българин отваря **квартални магазинчета** в **няколко града** и продава на **различни цени**:

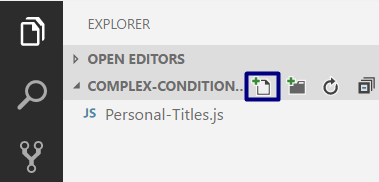
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| град / продукт | **coffee** | **water** | **beer** | **sweets** | **peanuts** |
| **Sofia** | 0.50 | 0.80 | 1.20 | 1.45 | 1.60 |
| **Plovdiv** | 0.40 | 0.70 | 1.15 | 1.30 | 1.50 |
| **Varna** | 0.45 | 0.70 | 1.10 | 1.35 | 1.55 |

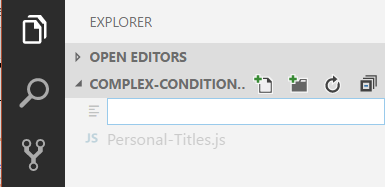
Напишете програма, която чете **град** (стринг), **продукт** (стринг) и **количество** (десетично число), въведени от потребителя, и пресмята и отпечатва **колко струва** съответното количество от избрания продукт в посочения град.

### Примерен вход и изход:

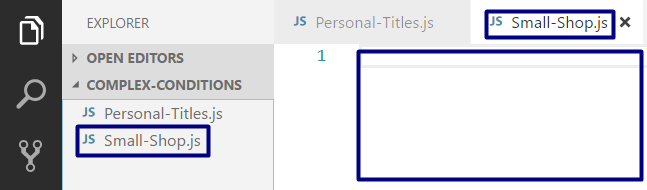
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| coffee  Varna  2 | 0.9 | peanuts  Plovdiv  1 | 1.5 | beer  Sofia  6 | 7.2 | water  Plovdiv  3 | 2.1 | sweets  Sofia  2.23 | 3.2335 |

1. Създайте **нов JavaScript файл** в съществуващата папка и го именувайте подходящо.





1. Съдържанието на новият файл ще се отвори в прозореца в дясно.



Вече имате нов JavaScript файл и остава да напишете кода за решаване на задачата.

1. Отидете във файла Small-Shop.js и създайте функцията smallShop([arg1, arg2, arg3]). Можете да си помогнете с картинката по-долу:

function smallShop([arg1, arg2, arg3]) {

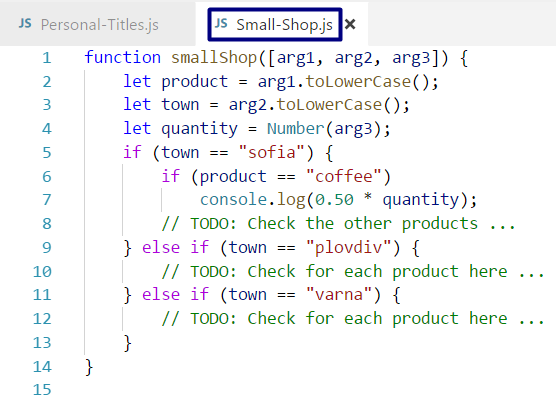
//Your code goes here...

}

//Call the function to execute the code inside

smallShop(["sweets", "Sofia", "2.23"]);

\* **Подсказка**: Можете да прехвърлите всички букви в долен регистър с .toLowerCase() за да сравнявате продукти и градове без значение на малки / главни букви:



1. Стартирайте програмата с Ctrl + [F5] и я тествайте с различни входни стойности.
2. **Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#1>.

## Точка в правоъгълник

Напишете програма, която проверява дали **точка {x, y}** се намира **вътре в правоъгълник {x1, y1} – {x2, y2}**. Входните данни се състоят от 6 аргумента(реда), въведени от потребителя: десетичните числа **x1**, **y1**, **x2**, **y2**, **x** и **y** (като се гарантира, че **x1 < x2** и **y1 < y2**). Една точка е вътрешна (“**Inside**”) за даден правоъгълник, ако се намира някъде във вътрешността му или върху някоя от страните му. Отпечатайте “**Inside**” или “**Outside**”.

### Примерен вход и изход:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **визуализация** |  | **вход** | **изход** | **визуализация** |
| 2  -3  12  3  8  -1 | Inside |  | 2  -3  12  3  11  -3.5 | Outside |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **визуализация** |  | **вход** | **изход** | **визуализация** |
| -1  -3  4  1  0.5  1 | Inside |  | -1  -3  4  1  -1.2  1.4 | Outside |  |

Използвайтескелета на функцията:

function pointInRectangle([arg1, arg2, arg3, arg4, arg5, arg6]) {

//Your code goes here ...

}

pointInRectangle(["2", "-3", "12", "3", "8", "-1"]);

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#2>.

\* **Подсказка**: една точка е вътрешна за даден многоъгълник, ако едновременно са изпълнени следните четири условия (можете да ги проверите с if проверка с логическо „**и**“ – оператор &&):

* Точката е надясно от лявата стена на правоъгълника (**x >= x1**)
* Точката е наляво от дясната стена на правоъгълника (**x <= x2**)
* Точката е надолу от горната стена на правоъгълника (**y >= y1**)
* Точката е нагоре от долната стена на правоъгълника (**y <= y2**)

## Плод или зеленчук?

Да се напише програма, която **чете име на продукт**, подадено от потребителя като аргумент, и проверява дали е **плод** или **зеленчук**.

* Плодовете "**fruit**" са **banana**, **apple**, **kiwi**, **cherry**, **lemon** и **grapes**
* Зеленчуците "**vegetable**" са **tomato**, **cucumber**, **pepper** и **carrot**
* Всички останали са "**unknown**"

Да се изведе “**fruit**”, “**vegetable**” или “**unknown**” според въведения продукт.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| banana | fruit | apple | fruit | tomato | vegetable | water | unknown |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#3>.

\* **Подсказка**: използвайте условна if проверка с логическо „**или**“ – operator ||.

## Невалидно число

Дадено **число е валидно**, ако е в диапазона [**100**…**200**] или е **0**. Да се напише програма, която **чете цяло число**, въведено от потребителя, и печата “**invalid**”, ако въведеното число **не е валидно**.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 75 | invalid | 150 | *(няма изход)* | 220 | invalid | 199 | *(няма изход)* |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| -1 | invalid | 100 | *(няма изход)* | 200 | *(няма изход)* | 0 | *(няма изход)* |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#4>.

\* **Подсказка**: използвайте условна if проверка с **отрицание** и логически операции.

## Точка върху страната на правоъгълник

Напишете програма, която проверява дали **точка {x, y}** се намира **върху някоя от страните на правоъгълник {x1, y1} – {x2, y2}**. Входните данни се състоят от 6 аргумента(реда), подадени от потребителя: числата **x1**, **y1**, **x2**, **y2**, **x** и **y** (като се гарантира, че **x1 < x2** и **y1 < y2**). Да се отпечата “Border” (точката лежи на някоя от страните) или “Inside / Outside” (в противен случай).

### Примерен вход и изход:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **визуализация** |  | **вход** | **изход** | **визуализация** |
| 2  -3  12  3  8  -1 | Inside / Outside |  | 2  -3  12  3  12  -1 | Border |  |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#5>.

\* **Подсказка**: използвайте една или няколко условни if проверки с логически операции. Точка **{x, y}** лежи върху някоя от страните на правоъгълник **{x1, y1} – {x2, y2}**, ако е изпълнено едно от следните условия:

* **x** съвпада с **x1** или **x2** и същевременно **y** е между **y1** и **y2**
* **y** съвпада с **y1** или **y2** и същевременно **x** е между **x1** и **x2**

Можете да проверите горните условия с една по-сложна if-else конструкция или с няколко по-прости проверки или с **вложени** if-else проверки.

## Магазин за плодове

Магазин за плодове през **работните дни** работи на следните **цени**:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **плод** | banana | apple | orange | grapefruit | kiwi | pineapple | grapes |
| **цена** | 2.50 | 1.20 | 0.85 | 1.45 | 2.70 | 5.50 | 3.85 |

В **събота** и **неделя** магазинът работи на **по-високи** **цени**:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **плод** | banana | apple | orange | grapefruit | kiwi | pineapple | grapes |
| **цена** | 2.70 | 1.25 | 0.90 | 1.60 | 3.00 | 5.60 | 4.20 |

Напишете програма, която чете **плод** (banana / apple / orange / grapefruit / kiwi / pineapple / grapes), **ден от седмицата** (Monday / Tuesday / Wednesday / Thursday / Friday / Saturday / Sunday) и **количество** (число), въведени от потребителя като аргументи, и пресмята **цената** според цените от таблиците по-горе. Резултатът да се отпечата **закръглен с 2 цифри** след десетичната точка. При невалиден ден от седмицата или невалидно име на плод да се отпечата “**error**”.

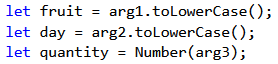
### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| apple  Tuesday  2 | 2.40 | orange  Sunday  3 | 2.70 | kiwi  Monday  2.5 | 6.75 | grapes  Saturday  0.5 | 2.10 | tomato  Monday  0.5 | error |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#6>.

\* **Подсказки**:

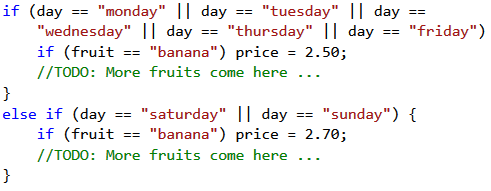
* Прочетете входа и обърнете името на плода и деня от седмицата в **малки букви**:



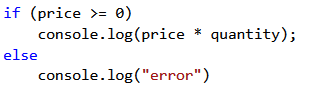
* Първоначално задайте цена **-1**:



* Използвайте вложени if проверки, за да изчислите цената за дадения плод и ден от седмицата:



* Накрая проверете цената. Ако все още е **-1**, значи даденият плод или денят от седмицата е **невалиден**. За да отпечатате точно **2 цифри след десетичната точка** (със закръгляне), използвайте форматиращ метод .toFixed(2). Кодът може да е подобен на следния:



## Търговски комисионни

Фирма дава следните **комисионни** на търговците си според **града**, в който работят и обема на **продажбите** s:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Град** | **0 ≤ s ≤ 500** | **500 < s ≤ 1 000** | **1 000 < s ≤ 10 000** | **s > 10 000** |
| Sofia | 5% | 7% | 8% | 12% |
| Varna | 4.5% | 7.5% | 10% | 13% |
| Plovdiv | 5.5% | 8% | 12% | 14.5% |

Напишете **конзолна програма**, която чете име на **град** (стринг) и обем на **продажби** (число), въведени от потребителя, и изчислява и извежда размера на търговската **комисионна** според горната таблица. Резултатът да се изведе закръглен с **2 цифри след десетичната точка**. При **невалиден** град или обем на продажбите (отрицателно число) да се отпечата “**error**”.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| Sofia  1500 | 120.00 | Plovdiv  499.99 | 27.50 | Varna  3874.50 | 387.45 | Kaspichan  -50 | error |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#7>.

\* **Подсказки**:

* Прочетете входа и **обърнете града в** **малки букви** (като в предходната задача).
* Първоначално задайте **комисионна -1**. Тя ще бъде променена, ако градът и ценовият диапазон бъдат намерени в таблицата с комисионните.
* Използвайте вложени if проверки, за **да изчислите комисионната** според града и според обема на продажбите. Може да си помогнете с кода по-долу:



* Накрая проверете комисионната. Ако все още е **-1**, значи въведеният град или обем продажби не се срещат в таблицата с комисионните и трябва да се отпечата “**error**”. В противен случай трябва да се изчисли комисионната (процент комисионна по обем на продажбите) и да се отпечата със закръгляне с точно **2 цифри след десетичната точка**. Може да използвате форматиращ метод .toFixed(2).

## Ден от седмицата

Напишете програма, която чете **цяло** **число**, въведено от потребителя, и отпечатва **ден от седмицата** (на английски език), в граници [1...7] или отпечатва “**Error**” в случай, че въведеното число е **невалидно**.

### Примерен вход и изход:

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 1 | Monday |
| 2 | Tuesday |
| 3 | Wednesday |
| 4 | Thursday |
| 5 | Friday |
| 6 | Saturday |
| 7 | Sunday |
| -1 | Error |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#8>.

**\*Подсказка:** Използвайте условната конструкция [**switch-case**](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/switch).

## Клас животно

Напишете програма, която отпечатва класа на животното според неговото име, въведено от потребителя.

* **dog -> mammal**
* **crocodile, tortoise, snake -> reptile**
* **others -> unknown**

### Примерен вход и изход:

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| dog | mammal |
| snake | reptile |
| cat | unknown |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#9>.

**\*Подсказка:** Използвайте условната конструкция switch-case.

## Кино

В една кинозала столовете са наредени в правоъгълна форма в r реда и c колони. Има три вида прожекции с билети на различни цени:

* Premiere – премиерна прожекция, на цена 12.00 лева.
* Normal – стандартна прожекция, на цена 7.50 лева.
* Discount – прожекция за деца, ученици и студенти на намалена цена от 5.00 лева.

Напишете програма, която чете **тип прожекция** (стринг), брой **редове** и брой **колони** в залата (цели числа), въведени от потребителя като аргументи, и изчислява общите приходи от билети при пълна зала. Резултатът да се отпечата във формат като в примерите по-долу, с 2 знака след десетичната точка.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| Premiere  10  12 | 1440.00 leva | Normal  21  13 | 2047.50 leva | Discount  12  30 | 1800.00 leva |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#10>.

\* **Подсказка**: използвайте прости проверки и елементарни изчисления. За да изведете резултата с точно 2 цифри след десетичната точка, използвайте форматиращ метод .toFixed(2).

## Волейбол

Влади е студент, живее в София и си ходи от време на време до родния град. Той е много запален по волейбола, но е зает през работните дни и играе **волейбол** само през **уикендите** и в **празничните дни**. Влади играе **в София** всяка **събота**, когато **не е на работа** и **не си пътува до родния град**, както и в **2/3 от празничните дни**. Той пътува до **родния си град** h **пъти** в годината, където играе волейбол със старите си приятели в **неделя**. Влади **не е на работа 3/4 от уикендите**, в които е в София.Отделно, през **високосните години** Влади играе с **15% повече** волейбол от нормалното. Приемаме, че годината има точно **48 уикенда**, подходящи за волейбол.

Напишете програма, която изчислява **колко пъти Влади е играл волейбол** през годината. **Закръглете резултата** надолу до най-близкото цяло число (например 2.15 🡪 2; 9.95 🡪 9).

Входните данни се въвеждат от потребителя като аргументи, в следния вид:

* Първият аргумент(ред) е думата “leap” (високосна година) или “normal” (невисокосна).
* Вторият аргумент(ред) е цялото число p – брой празници в годината (които не са събота и неделя).
* Третият аргумент(ред) е цялото число h – брой уикенди, в които Влади си пътува до родния град.

### Примерен вход и изход:

|  |  |  |
| --- | --- | --- |
| **вход** | **изход** | **Коментари** |
| leap  5  2 | 45 | 48 уикенда в годината, разделени по следния начин:   * 46 уикенда в София 🡪 46 \* 3 / 4 🡪 **34.5** съботни игри в София * 2 уикенда в родния си град 🡪 2 недели 🡪 **2** игри в неделя в родния град   5 празника:   * 5 \* 2/3 🡪 **3.333** игри в София в празничен ден   Общо игри през уикенди и празници в София и в родния град: 34.5 + 2 + 3.333 🡪 **39.833**  Годината е високосна:   * Влади играе допълнителни 15% \* 39.833 🡪 **5.975** игри волейбол   Общо игри през цялата година:   * 39.833 + 5.975 = **45.808** игри * Резултатът е **45** (закръгля се надолу) |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| normal  3  2 | 38 | leap  2  3 | 43 | normal  11  6 | 44 | leap  0  1 | 41 | normal  6  13 | 43 |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/1013#11>.

\* **Подсказки**:

* Пресметнете **уикендите в София** (48 минус уикендите в родния град). Пресметнете **броя игри в уикендите в София**: умножете уикендите в София с (3 / 4) или 0.75.
* Пресметнете **броя игри в родния град**. Те са точно колкото са пътуванията до родния град.
* Пресметнете **броя игри в празничен ден**. Те са броя празници умножени по (2 / 3).
* **Сумирайте** броя на всички игри. Той е дробно число. Не бързайте да закръгляте още.
* Ако годината е **високосна**, добавете **15%** към общия брой игри.
* Накрая **закръглете** надолу до най-близкото цяло число с Math.floor(result).

## \* Точка във фигурата

**Фигура** се състои от **6 блокчета** **с размер** **h \* h**, разположени като на фигурата вдясно. Долният ляв ъгъл на сградата е на позиция {0, 0}. Горният десен ъгъл на фигурата е на позиция {**2\*h**, **4\*h**}. На фигурата координатите са дадени при **h = 2**.

Напишете програма, която чете цяло число h и координатите на дадена **точка** {x, y} (цели числа), въведени от потребителя като аргументи, и отпечатва дали точката е вътре във фигурата (inside), вън от фигурата (outside) или на някоя от стените на фигурата (border).

### Примерен вход и изход:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **визуализация** |  | **вход** | **изход** | **визуализация** |
| 2  3  10 | outside |  | 15  13  55 | outside |  |
| 2  3  1 | inside | 15  29  37 | inside |
| 2  2  2 | border | 15  37  18 | outside |
| 2  6  0 | border | 15  -4  7 | outside |
| 2  0  6 | outside | 15  30  0 | border |

**Тествайте** решението си в **judge системата**: <https://judge.softuni.bg/Contests/Practice/Index/153#12>.

\* **Подсказки**:

* Може да разделите фигурата на **два правоъгълника** с обща стена.
* Една точка е **външна** (outside) за фигурата, когато е едновременно **извън** двата правоъгълника.
* Една точка е **вътрешна** (inside) за фигурата, ако е вътре в някой от правоъгълниците (изключвайки стените им) или лежи върху общата им стена.
* В **противен случай** точката лежи на стената на правоъгълника (border).

# Упражнения: Графични и Web приложения

## \* Точка и правоъгълник – графично (GUI) приложение

Да се разработи графично (**GUI**) приложение за **визуализация на точка и правоъгълник**. Приложението трябва да изглежда приблизително по следния начин:

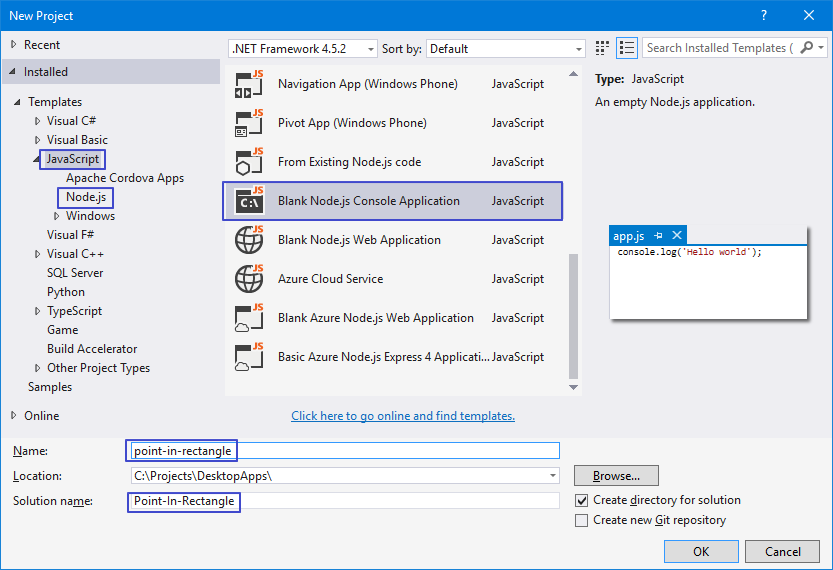


От контролите вляво се задават координатите на **два от ъглите на правоъгълник** (десетични числа) и координатите на **точка**. Приложението **визуализира графично** правоъгълника и точката и изписва дали точката е **вътре** в правоъгълника (**Inside**), **вън** от него (**Outside**) или на някоя от стените му (**Border**).

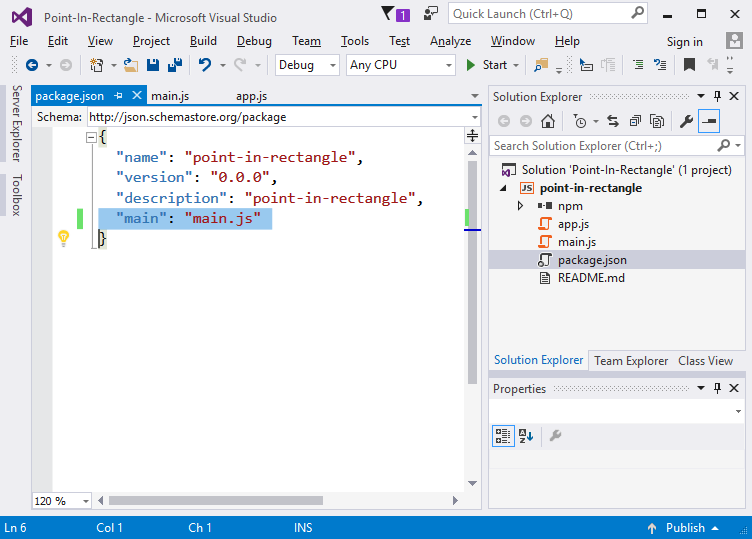
Приложението **мащабира** координатите на правоъгълника и точката, за да бъдат максимално големи, но да се събират в полето за визуализация в дясната страна на приложението.

**Внимание**: това приложение е значително **по-сложно** от предходните графични приложения, които разработвахте до сега, защото изисква ползване на функции за чертане и нетривиални изчисления за преоразмеряване и преместване на правоъгълника и точката. Следват инструкции за изграждане на приложението стъпка по стъпка.

1. За целта ще използваме [**Electron.js**](https://electron.atom.io/) framework. Създайте нов Visual Studio проект. Изберете от диалоговия прозорец [Templates] 🡪 [JavaScript] 🡪 [Node.js] 🡪 [**Blank Node.js Console Application**] и дайте подходящо име на проекта, например “Point-In-Rectangle”:



1. Добавете в проекта javascript файл с име "**main.js**". В случай, че не сте инсталирали Electron.js, преминете през стъпки **2.** и **3.** от Упражнение "**1. First steps in coding**", задача "**7. Десктоп приложение "Суматор за числа"** ".
2. Отворете файла "package.json", който съдържа настройките на проекта и променете името на стартовия скриптов файл с "main.js" :



1. Кодът, описан в "main.js", управлява събитията и създава нови прозорци в приложението. Трябва да изглежда по следния начин:

const {app, BrowserWindow} = require('electron');

const path = require('path');

const url = require('url');

let win;

function createWindow () {

win = new BrowserWindow({width: 750, height: 300, resizable: false});

win.loadURL(url.format({

pathname: path.join(\_\_dirname, 'index.html'),

protocol: 'file:',

slashes: true

}));

win.on('closed', ()=>{

win = null;

});

}

app.on('ready', createWindow);

app.on('window-all-closed', () => {

app.quit()

})

app.on('activate', () => {

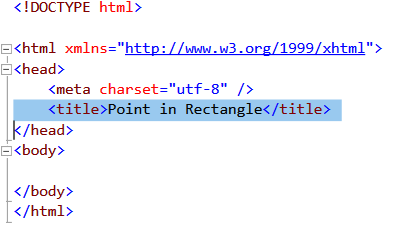
if (win === null) {

createWindow()

}

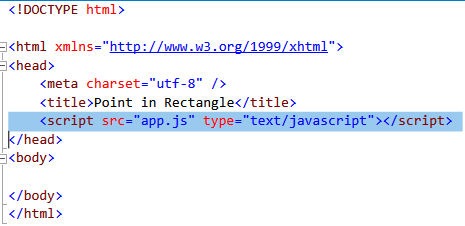
})

1. Създайте нов \*.html файл с име "**index.html**". Тагът "**<title>**" е задължителен за всеки html документ и дефинира заглавието му. Влезте в него и напишете "**Point in Rectangle**":



Добавете следния код под тага **<title>** в "index.html":

<script src="app.js" type="text/javascript"></script>



По този начин се осъществява връзката между файловете **index.html** и **app.js**.

Тагът "**<body>**" дефинира тялото на html документа. Напишете в него следния код:

<body>

<div style="float:left">

<br />

<label>Rectangle X1:</label>

<input id="rect-x1" type="number" />

<br />

<label>Rectangle Y1:</label>

<input id="rect-y1" type="number" />

<br />

<label>Rectangle X2:</label>

<input id="rect-x2" type="number" />

<br />

<label>Rectangle Y2:</label>

<input id="rect-y2" type="number" />

<br />

<label>Point X:</label>

<input id="point-x" type="number" />

<br />

<label>Point Y:</label>

<input id="point-y" type="number" />

<br />

<input type="button" onclick="draw()" value="Draw!" />

<br />

<div id="result">

<label>Status:</label>

<span id="status"></span>

</div>

</div>

<div style="float:right">

<canvas style="border: 2px solid orange;" id="a" width="400" height="200">

This text is displayed if your browser does not support HTML5 Canvas.

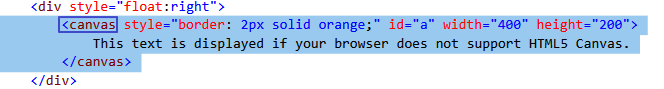
</canvas>

</div>

</body>

За въвеждане на координатите на правоъгълника и точката, използваме input полета от тип number, с надписи (**<label>**).

За да чертаем геометрични фигури в приложението, използваме html елемента [**<canvas>**](https://www.w3schools.com/html/html5_canvas.asp):

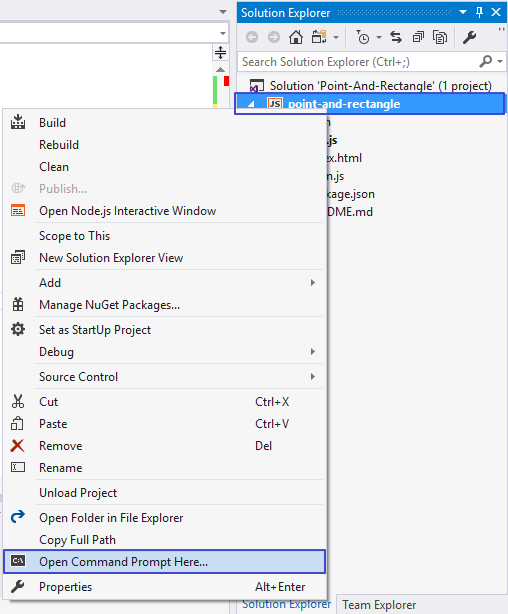


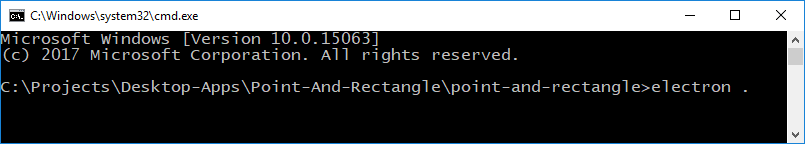
Той приема следните параметри:

* Ширина(width) в пиксели (px);
* Височина(height) в пиксели (px);
* Очертание(border);

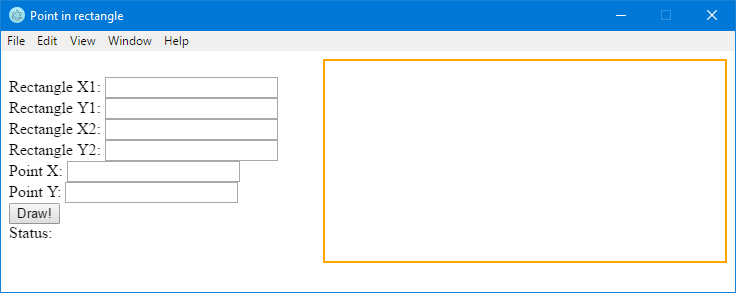
За да се отрази промяната в приложението, файловете трябва да се запазват с **[Ctrl+S]**.

Отворете **cmd** с десен клик на мишката върху проекта и стартирайте приложението с командата "**electron .**" :





**Приложението трябва да изглежда по следния начин:**



1. Остава да се имплементира най-сложната част: визуализация на правоъгълника и точката в полето на елемента <canvas>, чрез функцията draw() в app.js.

Създайте[**CanvasRenderingContext2D**](https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D)обекткато напишете следния код:

//Create canvas element

let canvas = document.getElementById('a');

let context = canvas.getContext('2d');

Елементът <canvas> е поле, в което обектът, генериран чрез метода .getContext('2d'), чертае графики, текст, изображения и други елементи. В случая променливата **context** представлява този обект.

Запишете в отделни променливи координатите на двата ъгъла на правоъгълника:

//Get input for rectangle coordinates

let rectX1 = Number(document.getElementById("rect-x1").value) \* 10;

let rectY1 = Number(document.getElementById("rect-y1").value) \* 10;

let rectX2 = Number(document.getElementById("rect-x2").value) \* 10;

let rectY2 = Number(document.getElementById("rect-y2").value) \* 10;

Стойностите на координатите са достъпни чрез **id** на **<input>** полетата. За по-добра визуализация на екрана, мащабирайте стойностите като **ги увеличите 10 пъти**.

Следващата стъпка е да се пресметнат страните на правоъгълника, тъй като обектът **context** рисува правоъгълник по четири параметъра – **x**-координатa и **y**-координата на горен ляв ъгъл, **ширина** в пиксели и **височина** в пиксели:

//Calculate rectangle parameters

let rectWidth = Math.abs(rectX1 - rectX2);

let rectHeight = Math.abs(rectY1 - rectY2);

Можете да ползвате **кода по-долу**, който рисува червен правоъгълник, според зададените във формата координати, използвайки метода [.strokeRect()](https://www.w3schools.com/tags/canvas_strokerect.asp):

//Set rectangle style

context.strokeStyle = "#ff0000";

context.lineWidth = 3;

//Draw rectangle with given parameters

context.strokeRect(rectX1, rectY1, rectWidth, rectHeight);

Аналогично на правоъгълника, взимаме координатите на точката и ги мащабираме. След това задаваме стил на точката – оранжев цвят. За по-добра визуализация на екрана, преобразувайте точката в кръг с метода .[arc](https://www.w3schools.com/tags/canvas_arc.asp)(). Този метод приема пет параметъра – **x**-координата, **y**-координата, **радиус**, **начало на дъгата** в радиани, **край на дъгата** в радиани:

//Get input for point coordinates

let pointX = Number(document.getElementById("point-x").value) \* 10;

let pointY = Number(document.getElementById("point-y").value) \* 10;

//Set point style and draw point

context.beginPath();

context.fillStyle = "#ffcc00";

context.arc(pointX, pointY, 4, 0, 2 \* Math.PI);

context.closePath();

context.fill();

За да отразите резултата от if-проверките, запазете в отделни променливи следните елементи от html кода:

//Assign variables to (<div id="result">) and (<span id="status">) html elements

let result = document.getElementById("status");

let output = document.getElementById("result");

Последната стъпка е проверката за позицията на точката спрямо правоъгълника:

//Check point position

if () {

result.innerHTML = "Inside";

output.style.backgroundColor = "palegreen";

}

else if () {

result.innerHTML = "Border";

output.style.backgroundColor = "gold";

}

else {

result.innerHTML = "Outside";

output.style.backgroundColor = "lightsalmon";

}

Помислете как **да допишете** недовършените (нарочно) условия в if-проверките! Кодът по-горе нарочно не се компилира, защото целта му е да помислите как и защо работи и да допишете липсващите части.

Горният код взима координатите на правоъгълника и точката и проверява дали точката е вътре, вън или на страната на правоъгълника. При визуализацията на резултата се сменя и цвета на фона на текстовия блок, който го съдържа.

1. Това е пълната версия на действието Draw():

function draw() {

//Create canvas element

let canvas = document.getElementById('a');

let context = canvas.getContext('2d');

//Clear canvas window

context.clearRect(0, 0, canvas.width, canvas.height);

//Get input for rectangle coordinates

let rectX1 = Number(document.getElementById("rect-x1").value) \* 10;

let rectY1 = Number(document.getElementById("rect-y1").value) \* 10;

let rectX2 = Number(document.getElementById("rect-x2").value) \* 10;

let rectY2 = Number(document.getElementById("rect-y2").value) \* 10;

//Calculate rectangle parameters

let rectWidth = Math.abs(rectX1 - rectX2);

let rectHeight = Math.abs(rectY1 - rectY2);

//Set rectangle style

context.strokeStyle = "#ff0000";

context.lineWidth = 3;

//Draw rectangle with given parameters

context.strokeRect(rectX1, rectY1, rectWidth, rectHeight);

//Get input for point coordinates

let pointX = Number(document.getElementById("point-x").value) \* 10;

let pointY = Number(document.getElementById("point-y").value) \* 10;

//Set point style and draw point

context.beginPath();

context.fillStyle = "#ffcc00";

context.arc(pointX, pointY, 4, 0, 2 \* Math.PI);

context.closePath();

context.fill();

//Assign variables to (div id="result") and (span id="status") html elements

let result = document.getElementById("status");

let output = document.getElementById("result");

//Check point position

if (pointX > rectX1 && pointX < rectX2 && pointY > rectY1 && pointY < rectY2) {

result.innerHTML = "Inside";

output.style.backgroundColor = "palegreen";

}

else if ((pointX == rectX1 || pointX == rectX2) && pointY > rectY1 && pointY < rectY2 ||

(pointY == rectY1 || pointY == rectY2) && pointX > rectX1 && pointX < rectX2) {

result.innerHTML = "Border";

output.style.backgroundColor = "gold";

}

else {

result.innerHTML = "Outside";

output.style.backgroundColor = "lightsalmon";

}

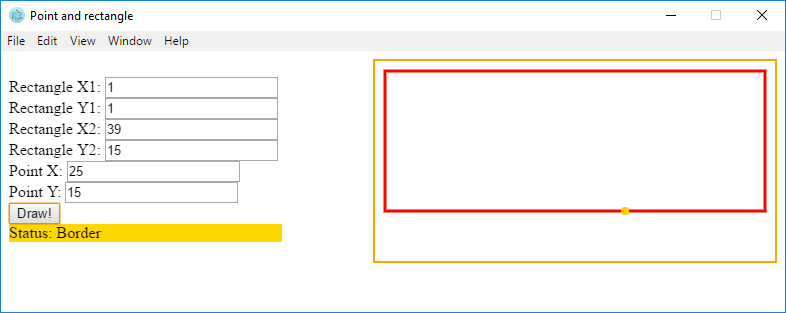
}

1. **Компилирайте кода**. Ако има някакви грешки, ги отстранете. Най-вероятната причина за грешка е ако сте написали кода на неправилно място.
2. **Стартирайте приложението** и го **тествайте** (с разцъкване). Пробвайте да въвеждате различни правоъгълници и позиционирайте точката на различни позиции, преоразмерявайте приложението и вижте дали се държи коректно.

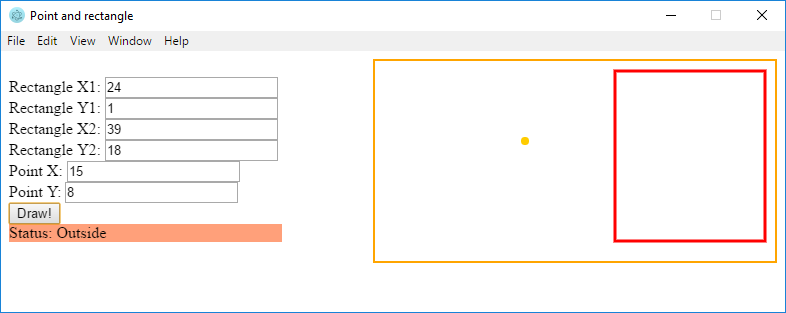
**Случай 1: Точката се намира в правоъгълника:**



**Случай 2: Точката лежи на една от страните на правоъгълника:**



**Случай 3: Точката се намира извън правоъгълника:**



# Изпитни задачи от минали издания на курса

## \* Навреме за изпит

*Трета задача от междинния изпит на 6 март 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/169#2)*.*

Студент трябва да отиде **на** **изпит** **в** **определен час** (например в 9:30 часа). Той идва в изпитната зала в даден **час на пристигане** (например 9:40). Счита се, че студентът е дошъл **навреме**, ако е пристигнал в часа на изпита или до половин час преди това. Ако е пристигнал по-рано повече от 30 минути, той е **подранил**. Ако е дошъл след часа на изпита, той е **закъснял**. Напишете програма, която прочита време на изпит и време на пристигане и отпечатва дали студентът е дошъл **навреме**, дали е **подранил** или е **закъснял** и **с колко часа или минути** е подранил или закъснял.

### Вход

Програмата чете **4 цели числа** (по едно на ред), въведени от потребителя:

* Първият ред(аргумент) съдържа **час на изпита** – цяло число от 0 до 23.
* Вторият ред(аргумент) съдържа **минута на изпита** – цяло число от 0 до 59.
* Третият ред(аргумент) съдържа **час на пристигане** – цяло число от 0 до 23.
* Четвъртият ред(аргумент) съдържа **минута на пристигане** – цяло число от 0 до 59.

### Изход

На първия ред отпечатайте:

* “**Late**”, ако студентът пристига по-късно от часа на изпита.
* “**On time**”, ако студентът пристига точно в часа на изпита или до 30 минути по-рано.
* “**Early**”, ако студентът пристига повече от 30 минути преди часа на изпита.

Ако студентът пристига с поне минута разлика от часа на изпита, отпечатайте на следващия ред:

* “mm **minutes before the start**” за идване по-рано с по-малко от час.
* “hh:mm **hours before the start**” за подраняване с 1 час или повече. Минутите винаги печатайте с 2 цифри, например “1:05”.
* “mm **minutes after the start**” за закъснение под час.
* “hh:mm **hours after the start**” за закъснение от 1 час или повече. Минутите винаги печатайте с 2 цифри, например “1:03”.

### Примерен вход и изход:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Вход** | **Изход** |  | **Вход** | **Изход** |  | **Вход** | **Изход** |
| 9  30  9  50 | Late  20 minutes after the start | 9  00  10  30 | Late  1:30 hours after the start | 10  00  10  00 | On time |
| 9  00  8  30 | On time  30 minutes before the start |  | 14  00  13  55 | On time  5 minutes before the start | 11  30  10  55 | Early  35 minutes before the start |
| 16  00  15  00 | Early  1:00 hours before the start |  | 11  30  8  12 | Early  3:18 hours before the start | 11  30  12  29 | Late  59 minutes after the start |

## \* Пътешествие

*Трета задача от изпита на 26 март 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Compete/Index/179#2)*.*

Странно, но повечето хора си плануват от рано почивката. Млад програмист разполага с **определен бюджет** и свободно време в даден **сезон**. Напишете програма, която да приема **на входа бюджета и сезона**, въведени от потребителя, а **на изхода** да изкарва, **къде ще почива** програмистът и **колко ще похарчи**.

**Бюджетът определя дестинацията, а** **сезонът определя колко от бюджета ще изхарчи**. Ако е **лято**, ще почива на **къмпинг**, а **зимата - в хотел**. Ако е в **Европа**, **независимо от сезона**, ще почива в **хотел**. Всеки **къмпинг** или **хотел**, **според дестинацията**, има **собствена цена**, която отговаря на даден **процент от бюджета**:

* При **100лв. или по-малко** – някъде в **България**
  + **Лято** – **30%** от бюджета
  + **Зима** – **70%** от бюджета
* При **1000лв. или по малко** – някъде на **Балканите**
  + **Лято** – **40%** от бюджета
  + **Зима** – **80%** от бюджета
* При **повече от 1000лв**. – някъде из **Европа**
  + При пътуване из Европа, независимо от сезона ще похарчи **90% от бюджета**.

### Вход

Входът се състои от **два реда(аргумента)**, въведени от потребителя:

* **Първи ред(аргумент)** – Бюджет, **реално число** в интервала **[10.00...5000.00].**
* **Втори ред(аргумент)** – Един от двата възможни сезона: **„summer”** или **“winter”**

### Изход

На конзолата трябва да се отпечатат **два реда**.

* **Първи ред** – „**Somewhere in [дестинация]**“ измежду “**Bulgaria**”,”**Balkans**” и ”**Europe**”
* **Втори ред** – “{**Вид почивка**} – {**Похарчена сума**}“
  + **Почивката** може да е между „**Camp**” и „**Hotel**”
  + **Сумата** трябва да е **закръглена с точност до вторият знак след запетаята**.

### Примерен вход и изход:

|  |  |
| --- | --- |
| **вход** | **изход** |
| 50  summer | Somewhere in Bulgaria  Camp - 15.00 |
| 75  winter | Somewhere in Bulgaria  Hotel - 52.50 |
| 312  summer | Somewhere in Balkans  Camp - 124.80 |
| 678.53  winter | Somewhere in Balkans  Hotel - 542.82 |
| 1500  summer | Somewhere in Europe  Hotel - 1350.00 |

## \* Операции между числа

*Трета задача от изпита на 24 април 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/181#2)*.*

Напишете програма, която чете **две цели числа (N1 и N2)** и **оператор**, с който да се **извърши** дадена **математическа операция** с тях. Възможните операции са: **Събиране(+)**, **Изваждане(-)**, **Умножение(\*)**, **Деление(/)** и **Модулно деление(%).** При **събиране**, **изваждане** и **умножение** на конзолата **трябва да се отпечатат резултатът** и дали той е **четен** или **нечетен**. При **обикновеното деление** – **резултатът**. При **модулното деление** – **остатъкът**. Трябва да се има предвид, че **делителят може да е равен на 0(нула)**, а **на нула не се дели**. В този случай трябва да се отпечата **специално съобщениe**.

### Вход

Програмата прочита **3 реда(аргумента)**, въведени от потребителя:

* **N1** – **цяло число** в интервала **[0...40 000]**
* **N2** – **цяло число** в интервала **[0...40 000]**
* **Оператор** – **един символ** измеду: „**+**“, „**-**“, „**\***“, „**/**“, „**%**“

### Изход

Да се отпечата на конзолата **един ред**:

* Ако операцията е **събиране**, **изваждене** или **умножение**:
  + „{N1} {оператор} {N2} = {резултат} – {even/odd}“
* Ако операцията е **деление**:
  + „{N1} / {N2} = {резултат}“ – резултатът е **фораматиран** до **втория знак след дес.запетая**
* Ако операцията е **модулно деление**:
  + „{N1} % {N2} = {остатък}“
* В случай на **деление с 0(нула)**:
  + „Cannot divide {N1} by zero“

### Примерен вход и изход:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **вход** | **изход** | **вход** | **изход** | **вход** | **изход** |
| 10  12  + | 10 + 12 = 22 - even | 123  12  / | 123 / 12 = 10.25 | 112  0  / | Cannot divide 112 by zero |
| 10  1  - | 10 – 1 = 9 - odd | 10  3  % | 10 % 3 = 1 | 10  0  % | Cannot divide 10 by zero |
| 7  3  \* | 7 \* 3 = 21 - odd |

## \* Билети за мач

*Трета задача от изпитa на 17 юли 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/233#2)*.*

Когато пуснали **билетите** **за Евро 2016**, **група** запалянковци **решили да си закупят**. **Билетите** имат **две категории с различни цени**:

* **VIP** – **499.99** лева.
* **Normal** – **249.99** лева.

Запалянковците **имат** **определен бюджет**, a **броят на хората** в групата **определя какъв процент от бюджета** трябва **да се задели за транспорт**:

* **От 1 до 4** – **75% от бюджета**.
* **От 5 до** **9** – **60% от бюджета**.
* **От 10 до 24** – **50% от бюджета**.
* **От 25 до 49** – **40% от бюджета**.
* **50 или повече** – **25% от бюджета**.

**Напишете програма**, която да **пресмята** **дали с останалите пари от бюджета** могат да си **купят билети за избраната категория**. И **колко пари** ще им **останат или ще са им нужни**.

### Вход

Програмата чете **точно 3 реда(аргумента)** , въведени от потребителя:

* На **първия** ред(аргумент) е **бюджетът** – **реално число в интервала [1 000.00 ... 1 000 000.00]**
* На **втория** ред(аргумент) е **категорията** – „**VIP**” или „**Normal**”
* На **третия** ред(аргумент) е **броят на хората в групата** – **цяло число в интервала [1 ... 200]**

### Изход

Да се **отпечата** на конзолата **един ред**:

* Ако **бюджетът е достатъчен**:
  + “**Yes! You have {N} leva left.**”– **N са останалите пари** на групата
* Ако **бюджетът НЕ Е достатъчен**:
  + “**Not enough money! You need {М} leva.**“ – където **М е сумата, която не достига**

**Сумите** трябва да са **форматирани с точност до два знака след десетичната запетая**.

### Примерен вход и изход:

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Вход** | **Изход** |
| 1000  Normal  1 | Yes! You have 0.01 leva left. | 30000  VIP  49 | Not enough money! You need 6499.51 leva. |
| **Обяснения** | | **Обяснения** | |
| **1 човек**: **75% от бюджета** отиват **за** **транспорт**  **Остават:** 1000 – 750 **= 250**  Категория **Normal**: билетът **струва 249.99 \* 1**  **249.99 < 250**: **остават** му 250 – 249.99 = **0.01** | | **49 човек**: **40% от бюджета** отиват **за** **транспорт**  **Остават:** 30000 – 12000 **= 18000**  Категория VIP: билета(ът) **струва 499.99 \* 49**  **24499.510000000002 < 18000**  **Не стигат** 24499.510000000002 - 18000 = **6499.51** | |

## \* Хотелска стая

*Трета задача от изпита на 28 август 2016. Тествайте решението си* [***тук***](https://judge.softuni.bg/Contests/Practice/Index/274#2)*.*

Хотел предлага **2 вида стаи**: **студио и апартамент**. Напишете програма, която изчислява **цената за целия престой за студио и апартамент**. **Цените** зависят от **месеца** на престоя:

|  |  |  |
| --- | --- | --- |
| **Май и октомври** | **Юни и септември** | **Юли и август** |
| Студио – **50** лв./нощувка | Студио – **75.20** лв./нощувка | Студио – **76** лв./нощувка |
| Апартамент – **65** лв./нощувка | Апартамент – **68.70** лв./нощувка | Апартамент – **77** лв./нощувка |

Предлагат се и следните **отстъпки**:

* За **студио**, при **повече** от **7** нощувки през **май и октомври** : **5% намаление**.
* За **студио**, при **повече** от **14** нощувки през **май и октомври** : **30% намаление**.
* За **студио**, при **повече** от **14** нощувки през **юни и септември**: **20% намаление**.
* За **апартамент**, при **повече** от **14** нощувки**, без значение от месеца : 10% намаление.**

### Вход

Програмата чете **точно 2 реда(аргумента)**, въведени от потребителя:

* На **първия** ред(аргумент) е **месецът** – **May,** **June,** **July,** **August,** **September** или **October**
* На **втория** ред(аргумент) е **броят на нощувките** – **цяло число в интервала [0 ... 200]**

### Изход

Да се **отпечатат** на конзолата **2 реда**:

* На **първия ред**: “**Apartment: {цена за целият престой} lv.**”
* На **втория ред**: “**Studio: {цена за целият престой} lv.**“

**Цената за целия престой трябва да се** **форматира с точност до два знака след десетичната запетая**.

### Примерен вход и изход:

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** | |
| May  15 | Apartment: 877.50 lv.  Studio: 525.00 lv. | **През май**, при повече от **14** нощувки, намаляваме цената на студиото с **30%** (50 – 15 = 35), а на апартамента – с **10%** (65 – 6.5 =58.5).  Целият престой в **апартамент** – **877.50 лв**.  Целият престой в **студио** – **525.00 лв**. | |
| **Вход** | **Изход** | **Вход** | **Изход** |
| June  14 | Apartment: 961.80 lv.  Studio: 1052.80 lv. | August  20 | Apartment: 1386.00 lv.  Studio: 1520.00 lv. |