АЛГОРИТАМСКИ НАЧИН РАЗМИШЉАЊА

У свету у коме живимо програмирање треба посматрати као продужетак писања. Програмирајући, можемо да стварамо интерактивне приче, симулације и анимације. Алгоритам је низ корака којих се морамо придржавати како би дошли до решења неког задатка.

Постоје три типа алгоритма:

1) **Линијски алгоритам** - извршава се корак по корак, од првог до последњег, и сваки од корака се извршава тачно једанпут;
2) **Циклични алгоритми** - у њима примећујемо понављања;
3) **Условни (разгранати) алгоритми** - то су решења задатака који зависе од неког услова. У условним алгоритмима неке наредбе ће се извршити, а неке неће, у зависности од испуњености услова.

Погледај видео лекцији Лекција 1 – Алгоритамски начин размишљања, на адреси: https://youtu.be/rETAQ5uYbWE?list=PLvvY5P8IMAsA-KcZDwMTbhUKWg2ircoGO

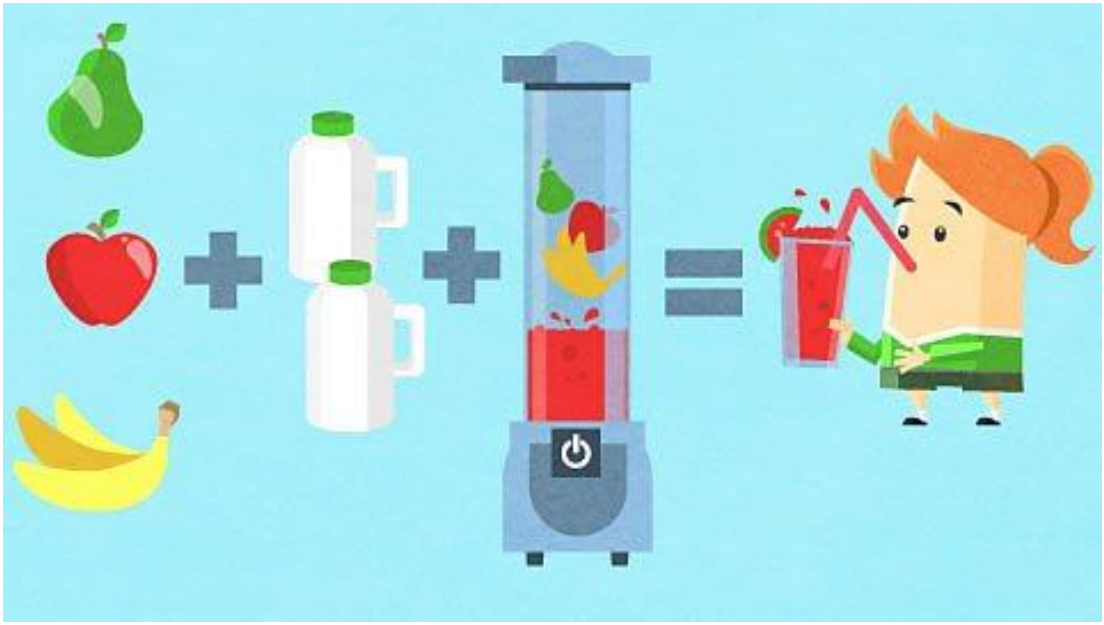
Шта мислиш, који тип алгоритма користи Ксенија да би избегла шетање Болета? Покушај да објасниш свој став.

Слика 1. Ксенија и њен алгоритам за шетање пса

Није на одмет да знаш више о типовима алгоритама.

Алгоритми код којих се сваки корак (инструкција) извршава једном по реду који је наведен називају се линијски алгоритми.

На пример, да бисте себи направи моћан воћни шејк морате пратити низ корака у одговарајућем редоследу. Ако радите нешто погрешно направићете неред у кухињи, или нећете добити воћни шејк.



Слика 2. Креирање моћног воћног шејка

У случају прављења шејка наш алгоритам је јасан сет (низ) инструкција.

Упутство за прављење воћног шејка:

1) Додајте воће у блендер.
2) Додати млеко у блендер.
3) Ставите поклопац на блендеру.
4) Укључите блендер.

Шта би се догодило ако бисте неки корак изоставили, или им обрнули редослед?

Могли сте да завршите поступак за прављење шејка са укључивањем блендера у коме нема воћа.

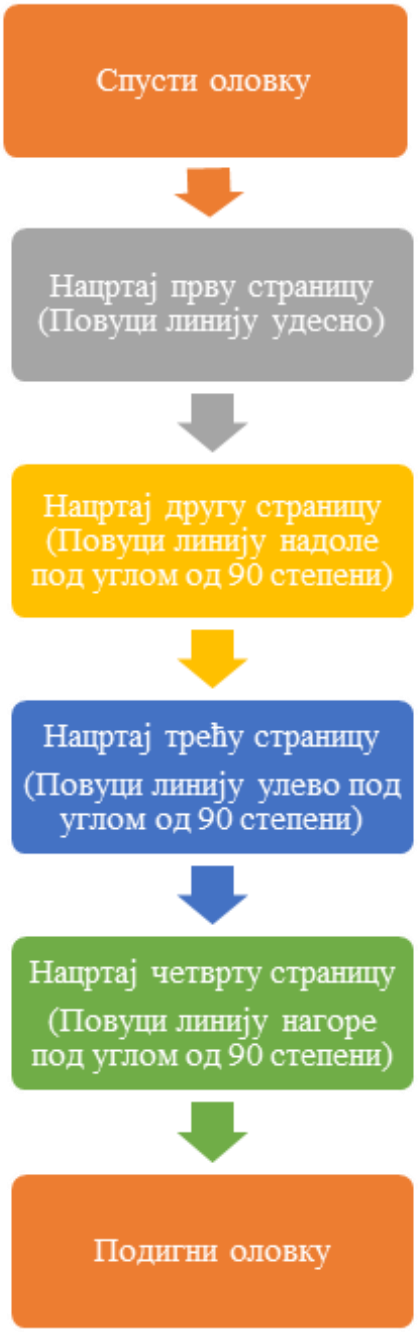
Или, могли сте да сипате само млеко, без воћа. То онда не би био воћни шејк, зар не? У ствари, то не би био шејк уопште!

А, шта би се десило да сте заборавили да ставити поклопац? Неред!

Свако од нас може решити одређени проблем на свој начин, тачније може постојати више алгоритама за решавање истог проблема.

Да ли се сећаш када ти је учитељица давала задатак да нацрташ квадрат? Опиши поступак решавања.

Један алгоритам за цртање квадрата може изгледати овако:

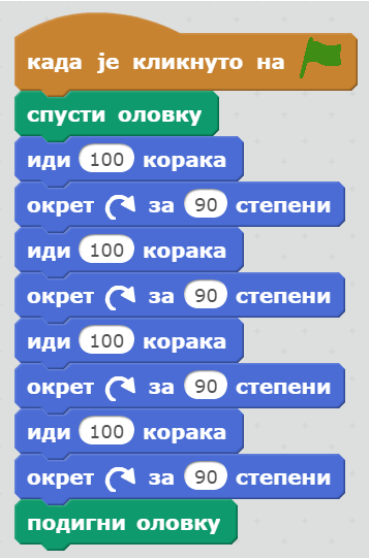


Слика 3. Приказ корака за исцртавање квадрата

Колико још различитих решења (агоритама) има за исцртавање квадрата? Пробај да даш своје решење.

Оно што је најзанимљивије јесте да сваки алгоритам, за било који проблем, може да се користи у програмирању за стварање најразличитијих програма (апликација) у програмском језику Скреч.

Показаћемо ти како изгледа алгоритам за исцртавање квадрата приказан у Скречу.



Слика 4. Приказ корака за исцртавање квадрата

Помоћ при изради овог задатка можеш наћи у видео лекцији Лекција 5 – Корњача графика, на адреси: https://www.youtube.com/edit?o=U&video_id=B70TOn8wVWM

Да бисмо научили неку песмицу, морамо је поновити неколико пута. Колико вам је потребно понављања неке песмице да бисте је научили. Два пута, три пута, ... десет пута?

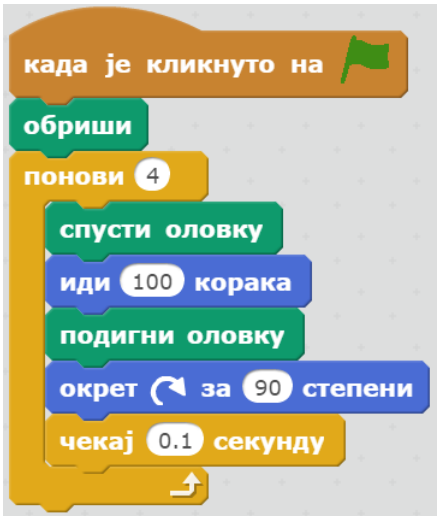
Као и у стварном животу, понављања (циклуси) се могу применити и у програмирању. Понављања су саставни део скоро сваког програма.

Алгоритми код којих се поједини кораци понављају више пута називају се циклични алгоритми. Када смо причали о линијским алгоритмима помињали смо кораке које треба да урадимо да бисмо нацртали квадрат. Сећате се да смо имали 8 корака. Овај проблем бисмо могли решити понављањем два иста корака:

1) Нацртај страницу
2) Окрени се за 90 степени

четири пута.

Овај проблем приказан у Скречу изгледа овако:



Слика 5. Изглед програма за цртање квадрата

Горњи пример је илустрација понављања за тачно одређени број пута (у нашем случају 4 пута). Осим оваквих врста понављања, некада је потребно да имамо кораке који се непрестано понављају. О томе ћемо говорити касније.

Покушај да напишеш алгоритам за цртање степеница.

Упореди своје решење са решењем датим у видео лекцији Лекција 5 – Корњача графика, на адреси: https://www.youtube.com/edit?o=U&video_id=B70TOn8wVWM

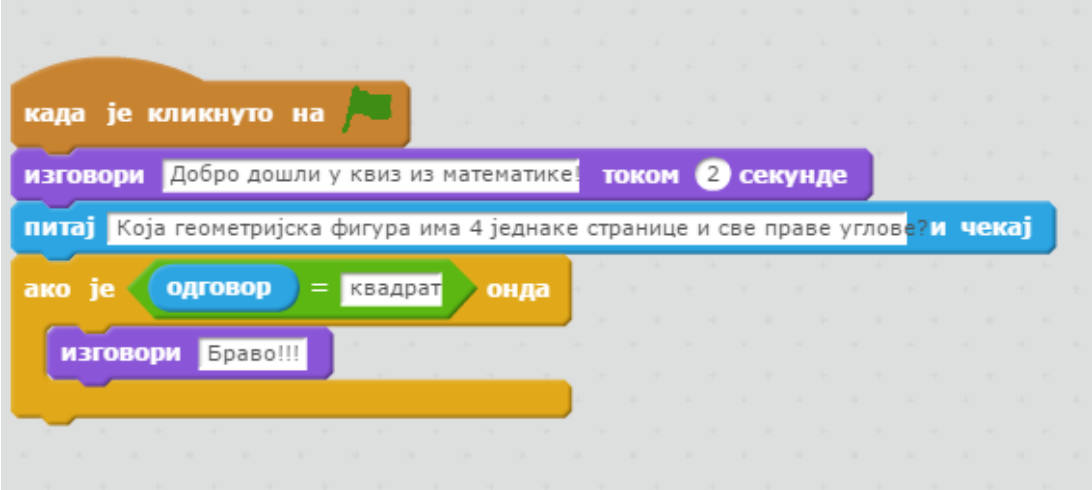
Колико пута сте се до сада сретали са неким условима у свом животу? Ако будеш одличан купићу ти таблет, или ако буде сунчано за викенд идемо на базен.

Као и у стварном животу услови се могу применити и у програмирању и они су саставни део скоро сваког програма. За разлику од линијских алгоритама где се сваки корак, односно свака наредба, извршава само једанпут, у условном алгоритму неке наредбе ће се извршити, а неке не, у зависности од испуњености услова.

Замислите да се такмичите у квизу из математике. Један од могућих сценарија би изгледао овако:

1) Водитељ ти пожели добродошлицу;
2) Пита „ Која геометријска фигура има 4 једнаке странице и све праве углове?“;
3) Ако одговориш „ квадрат“ водитељ те похвали.

Овај проблем, решен у Скречу, могао би да изгледа овако:



Слика 6. Изглед дела кода

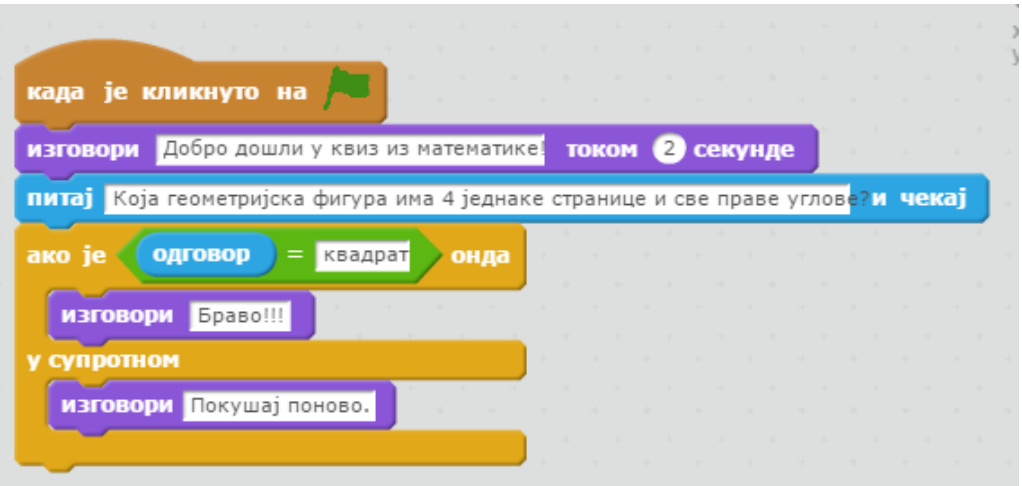
У овом случају, похвала за тачан одговор је загарантована. Али, шта би се догодило у случају погрешног одговора?

У праву си – ништа се не би догодило! Не бисмо добили информацију о погрешаном одговору. Имаш ли идеју како бисмо могли да изменимо наш програм тако да нам даје повратну информацију било да смо одговорили тачно, било нетачно? Напиши алгоритам.

Ми смо размишљали овако:

1) Водитељ ти пожели добродошлицу;
2) Пита „Која геометријска фигура има 4 једнаке странице и све праве углове?“;
3) Ако одговориш „ квадрат“ водитељ те похвали.
4) Ако одговориш нешто друго, водитељ ти каже да покушаш поново.

Овај проблем приказан у Скречу могао би да изгледа овако:



Слика 7. Изглед функционалнијег програма

Приказано решење је дефинитивно функционалније. Сада имаш обезбеђену повратну информацију и када одговориш тачно и када погрешиш.

Задатак: Покушај да направиш алгоритам за квиз који има три питања.