МЕТОДИЧЕСКИ АСПЕКТИ ЗА ФОРМИРАНЕ НА РЕФЛЕКСИВНИ УМЕНИЯ ПРИ ИЗУЧАВАНЕ НА СКРИПТОВИ ЕЗИЦИ В ПРОГИМНАЗИЯТА

Стефка Анева[1], Елена Тодорова[2, *]

[1, 2] Факултет по математика и информатика,
ПУ „Паисий Хилендарски“, Пловдив, бул. „България“ № 236

[1] stfaneva@uni-plovdiv.bg

[2] Автор за кореспонденция: etodorova@uni-plovdiv.bg*

Резюме. В настоящата статия се представя един подход, свързан с изучаването на скриптови текстови езици в прогимназията и целенасочено осъществяване на рефлексия в процеса на обучението по предмета „Компютърно моделиране и информационни технологии“. Предложени са набор от задачи и техни конкретни реализации чрез езика Python, свързани с развитие на алгоритмични знания и формиране на рефлексивни умения у учениците.

Ключови думи: *методика на обучение, алгоритмични умения, рефлексия, компютърно моделиране, програмиране, скриптов текстов език*

Въведение

В съвременното образование все по-голямо значение се отдава както на процеса на усвояване на концептуални знания, свързани с изучаването на даден учебен предмет, така и на формирането на умения за критично мислене, рефлексия и саморефлексия. Ключова роля в случая играе учителят, който трябва да създаде подходящи условия и учебни ситуации, провокиращи мисловна активност и самонаблюдение в процеса на обучението. Целенасочената педагогическа подкрепа, съчетана със системното прилагане на рефлексивни практики насочват обучаемите към осъзнаване на нуждата от личен опит, самостоятелно познание, както и към изграждане на нагласа за учене през целия живот.

Някои автори в своите изследвания дискутират въпроса за формиране на рефлексивни умения с фокус в обучението по математика, информатика

и информационни технологии [1, 2, 3, 4 и др.]. В контекста на обучението по „Компютърно моделиране и информационни технологии“ (КМИТ) в 5. – 7. клас, тематиката, свързана с програмиране със скриптови езици, предоставя благоприятна среда за развитие както на алгоритмично мислене, така и на рефлексивни способности. Скриптовите езици са програмни езици, които използват интерпретатор при изпълнение на програмен код – предимство, което ги прави бързи за разработка и лесни за тестване.

Практически задачи по програмиране в прогимназията, свързани с формиране на рефлексивни умения у учениците

В разработката представяме методически подход за формиране на рефлексивни способности у учениците в обучението по „Компютърно моделиране и информационни технологии“ в 6.–7. клас при изучаване на темата „Компютърно моделиране“. Акцентът при изучаване на темата в 6. клас е свързан със запознаване с конкретен скриптов текстов език за програмиране и създаване на интерактивни приложения с него. В контекста на учебното съдържание се разглеждат задачи, свързани с реализация на различни алгоритми, използвайки линейни, разклонени или циклични програмни конструкции; задачи за изчертаване на геометрични обекти, както и задачи за реализиране на анимации [5]. При изучаване на темата „Компютърно моделиране“ в 7. клас акцентът е насочен към: обсъждане и създаване на програми, които моделират реални ситуации с използване на различни типове данни; правилен подбор на типа на данни, съобразно изискванията на конкретна задача; анализиране на условия на задачи, включващи повтарящи се действия; прилагане на подходящи оператори за цикъл при реализиране на алгоритъм за решаване на дадена задача [6].

Всяка учебна задача в процеса на обучение спомага както за формиране на нови знания, така и за затвърждаване на вече придобити умения на учениците, свързани с конкретни тематични аспекти от учебното съдържание. По този начин се формират умения за учене чрез опита, които са тясно свързани с уменията на учащите да осъществяват рефлексия, т.е. да разсъждават върху извършената от тях дейност [3].

В настоящата статия са предложени набор от седем задачи, с нарастваща степен на сложност, реализирани чрез езика Python. Те са съобразени с учебното съдържание по предмета КМИТ и могат да бъдат използвани в задължителната подготовка както в 6., така и за 7. клас. Тематичното съдържание на задачите е свързано с използване на костенуркова графика за изчертаване на композиции от геометрични фигури чрез прилагане на циклични алгоритмични конструкции. Първата задача се разглежда като базова, а останалите шест са до голяма степен тематично свързани помежду си, като за тяхното решаване се използват

предходни знания. При някои от задачите се разглеждат различни аспекти и последователност на изчертаване на посочените графични композиции. Предложеният набор от задачи способства за развитие на алгоритмичните знания и опит на учениците в областта на програмирането с използване на скриптов език Python. Този опит може да бъде свързан със следните дейности: умело използване на основните възможности и функционалности на модула *turtle*; дефиниране и използване на потребителски подпрограми с параметри; правилен подбор и прилагане на подходящи оператори за цикъл; конструиране на циклични алгоритми и използване на вложени циклични конструкции при решаване на даден проблем. Същевременно, паралелно с усвояването на технически знания, тези задачи могат да подпомогнат изграждането на способности за осъществяване на рефлексия и стимулирането на рефлексивно мислене в обучението по КМИТ.

Задачите с визуален резултат (графика) стимулират дискусия, рефлексия и саморефлексия. За всяка една от задачите учителят разисква нейното условие и представя нагледно пред учениците начина на изчертаване на съответната композиция от фигури чрез визуална демонстрация на самото изпълнение с помощта на конкретна програмна среда. По този начин учениците ще осмислят и разберат по-добре съответната специфика и последователност при изчертаване на включените елементи в графичната композиция, както и особеностите, с които трябва да се съобразят при конструиране на самия алгоритъм на решение.

Базова задача. Създайте програма на Python, която изчертава пет равностранни триъгълника с една и съща дължина на страната, разположени последователно в хоризонтална редица един до друг (Фиг. 1).

Методически коментари:

При обсъждане на задачата се дискутират основните етапи на нейното решение: дефиниране на подпрограма за изчертаване на равностранен триъгълник с един параметър (*дължина на страната*); реализиране на цикъл с краен брой повторения (цикъл *for*) за изчертаване на желаната композиция от триъгълници.

Програмен код на Python	Резултат

`import turtle`
`pen=turtle.Turtle()`
`pen.shape("triangle")`
`pen.pensize(3)`
`pen.speed(5)`
`pen.color("orange")`
`def draw_triangle(a):`
`  for i in range(3):`
`  pen.forward(a)`
	
	Фиг. 1

```
    pen.left(120)
a=30
br=5
for i in range(br):
  draw_triangle(a)
  pen.forward(a)
```

Задача 1. Създайте програма на Python, която изчертава краен брой равностранни триъгълници с една и съща дължина на страната, разположени последователно един до друг в кръг (Фиг. 2).

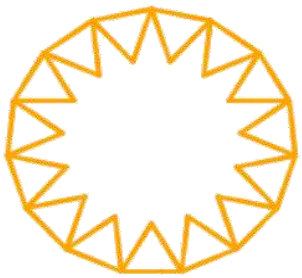
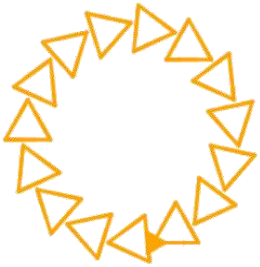
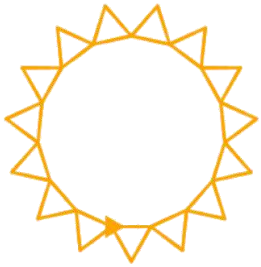
Методически коментари:

С учениците се коментира, че за реализирането на тази задача може да се използва програмния код на решението на базовата задача. За целта, в края на цикъла *for* трябва да се добави програмен код, реализиращ завъртането на костенурката наляво на определен ъгъл (*360/br*), осигуряващ подреждането на зададения краен брой (*br*) триъгълници в кръг.

Задача 2. Създайте програма на Python, която изчертава краен брой равностранни триъгълници с една и съща дължина на страната, разположени в кръг (Фиг. 3).

Методически коментари:

Обсъжда се разликата в начертаната композиция от Фиг. 3 спрямо предходната задача. В конкретния случай подредените триъгълници в кръг нямат допирни върхове. За постигане на този резултат е необходимо програмната инструкция *pen.left(360/br)* да се премести непосредствено след извикването на подпрограмата *draw_triangle(a)* за изчертаване на равностранен триъгълник. Обсъжда се необходимостта от използване на инструкциите *pen.penup()* и *pen.pendown()* при подготовката за изчертаване на следващата фигура.

Резултат и програмен код за зад. 1	**Резултат и програмен код за зад. 2**	**Резултат и програмен код за зад. 3**
 ***Фиг. 2***	 ***Фиг. 3***	 ***Фиг. 4***

`import turtle`
`pen=turtle.Turtle()`
`pen.shape("triangle")`
	`import turtle`
	`pen=turtle.Turtle()`
	`pen.shape("triangle")`
		`import turtle`
		`pen=turtle.Turtle()`
		`pen.shape("triangle")`

```
pen.pensize(3)
pen.speed(5)
pen.color("orange")
def draw_triangle(a):
  for i in range(3):
    pen.forward(a)
    pen.left(120)
a=30
br=15
for i in range(br):
  draw_triangle(a)
  pen.forward(a)
  pen.left(360/br)
```

```
pen.pensize(3)
pen.speed(5)
pen.color("orange")
def draw_triangle(a):
  for i in range(3):
    pen.forward(a)
    pen.left(120)
a=30
br=15
for i in range(br):
  draw_triangle(a)
  pen.left(360/br)
  pen.penup()
  pen.forward(a)
  pen.pendown()
```

```
pen.pensize(3)
pen.speed(5)
pen.color("orange")
def draw_triangle(a):
  for i in range(3):
    pen.forward(a)
    pen.right(120)
a=30
br=15
for i in range(br):
  draw_triangle(a)
  pen.forward(a)
  pen.left(360/br)
```

Задача 3. Създайте програма на Python, която изчертава краен брой равностранни триъгълници с една и съща дължина на страната, така че получената композиция да наподобява „слънце“ (Фиг. 4).

Вниманието на учениците се насочва към използване на решението на *задача 1,* така че с възможно най-малко корекции в програмния ѝ код да се получи композицията от Фиг. 4.

Методически коментари:

В случая се забелязва, че при новата композиция (Фиг. 4), триъгълниците са обърнати „навън“. За постигането на този резултат, в подпрограмата за изчертаване на равностранен триъгълник се извършва корекция, касаеща посоката на завъртане на 120 градуса вместо наляво, да бъде надясно – *pen.right(120)*.

Задача 4. Създайте програма на Python, която изчертава представената на Фиг. 5 композиция, съчетаваща две последователности от краен брой равностранни триъгълници с една и съща дължина на страната, разположени в кръг. Двата отделни „кръга“ от триъгълници (външен и вътрешен) да се изчертават последователно и да бъдат запълнени в два различни цвята.

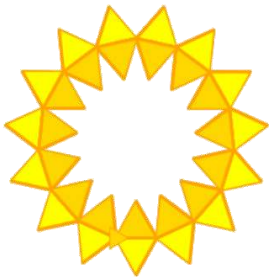
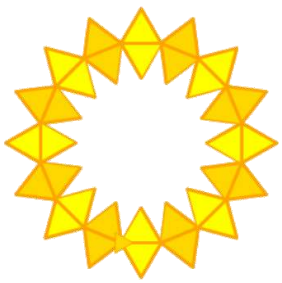
Методически коментари:

Основните акценти при решаването на задачата се свеждат до следното: дефиниране на две отделни подпрограми за изчертаване на равностранен триъгълник, различаващи се по посоката на завъртане на 120 градуса (надясно или наляво); използване на два отделни цикъла *for* за изчертаване на външната и вътрешната композиция от триъгълници, всяка от тях запълнена с различен цвят.

Задача 5. Създайте програма на Python, която изчертава представената на Фиг. 6 композиция от краен брой отделни елементи (*всеки елемент се състои от два триъгълника с обща страна*), така че всеки от тях да бъде запълнен с различен цвят, като се редуват алтернативно два избрани цвята.

Методически коментари:

Разликата спрямо задача 4 се състои в последователността на изчертаване на елементите на композицията (*с единичен елемент, включващ двата триъгълника с обща страна*) и в алтернативно редуване на два цвята на запълване при изчертаване на всеки съставен елемент. При решаване на задачата отново ще се използват две отделни подпрограми за изчертаване на триъгълник. За да се осъществи по-лесно алтернативното редуване на два цвята за запълване ще се дефинира списък *colors*, съдържащ два елемента – *"yellow"* и *"gold"*. Особеност в задачата е, че броя на елементите в композицията трябва да бъде кратен на дължината на списъка с цветове, за да се получи точен брой повторения на избраната „шарка“ за запълване на съответните елементи с редуващи се цветове. В представената композиция на Фиг. 6 се използват два цвята за запълване (*т.е. списъкът colors съдържа два елемента*), които се редуват последователно при изчертаването на включените 16 елемента в комплексната фигура (*т.е. в случая съставните елементи трябва да бъдат четен брой*). С помощта на вложени цикли *for* ще се извърши изчертаването на съответния брой елементи, запълнени последователно в два цвята. Външният цикъл *for* указва колко пъти ще се повтори „комбинацията“ от двойка елементи с различно запълване, а вътрешния цикъл осигурява самата смяна на желания цвят чрез последователно обхождане на елементите от списъка с цветове. В случая са налице 8 повторения на избраната комбинация от двойка елементи, запълнени с различен цвят. Допълнително към задачата може да се реализира възможността за директно въвеждане на броя елементи с контрол на входа, осигуряващ проверка за кратност на въведената стойност спрямо броя на елементи в списъка с цветове.

Резултат и програмен код за зад. 4	**Резултат и програмен код за зад. 5**
 ***Фиг. 5***	 ***Фиг. 6***
`import turtle`	`import turtle`

```
pen=turtle.Turtle()
pen.shape("triangle")
pen.pensize(3)
pen.speed(5)
pen.color("orange")
def draw_triangle1(a):
  for i in range(3):
    pen.forward(a)
    pen.right(120)
def draw_triangle2(a):
  for i in range(3):
    pen.forward(a)
    pen.left(120)
a=30
br=15
pen.fillcolor("yellow")
for i in range(br):
  pen.begin_fill()
  draw_triangle1(a)
  pen.end_fill()
  pen.forward(a)
  pen.left(360/br)
pen.fillcolor("gold")
for i in range(br):
  pen.begin_fill()
  draw_triangle2(a)
  pen.end_fill()
  pen.forward(a)
  pen.left(360/br)
```

```
pen=turtle.Turtle()
pen.shape("triangle")
pen.pensize(3)
pen.speed(5)
pen.color("orange")
colors=["yellow","gold"]
def draw_triangle1(a):
  for i in range(3):
    pen.forward(a)
    pen.right(120)
def draw_triangle2(a):
  for i in range(3):
    pen.forward(a)
    pen.left(120)
a=30
br=16
for i in range(int(br/len(colors))):
  for c in colors:
    pen.fillcolor(c)
    pen.begin_fill()
    draw_triangle1(a)
    draw_triangle2(a)
    pen.end_fill()
    pen.forward(a)
    pen.left(360/br)
```

Задача 6. Модифицирайте създадената програма на Python в зад. 5, като вместо използваните в нея две отделни подпрограми дефинирате само една подпрограма за изчертаване на равностранен триъгълник с два параметъра – страна на триъгълника и посока на завъртане. Представеният резултат на Фиг. 7 е аналогичен на този от Фиг. 6.

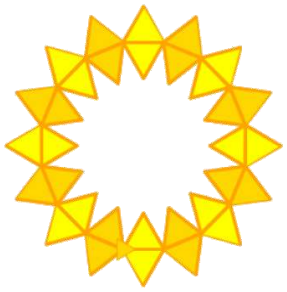
Задача 7. Създайте програма на Python, която изчертава представената на Фиг. 8 композиция от краен брой **ромбове** с остър ъгъл 60 градуса, така че всяка отделна фигура (*елемент от композицията*) да бъде с една и съща дължина на страната и запълнена с различен цвят, като се редуват алтернативно два избрани цвята.

Методически коментари:

В случая ще се дефинира подпрограма за изчертаване на ромб с два параметъра – страна и ъгъл. Аналогично, както в зад. 5, и тук броя на

елементите в композицията трябва да бъде кратен на дължината на дефинирания списък с цветове.

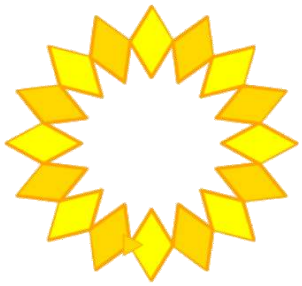
Резултат и програмен код за зад. 6



Фиг. 7

```
import turtle
pen=turtle.Turtle()
pen.shape("triangle")
pen.pensize(3)
pen.speed(5)
pen.color("orange")
colors=["yellow","gold"]
def draw_triangle(a,posoka):
  for i in range(3):
    pen.forward(a)
    if posoka=="left":
      pen.right(120)
    else:
      pen.left(120)
a=30
br=16
for i in range(int(br/len(colors))):
  for c in colors:
    pen.fillcolor(c)
    pen.begin_fill()
    draw_triangle(a,"right")
    draw_triangle(a,"left")
    pen.end_fill()
    pen.forward(a)
    pen.left(360/br)
```

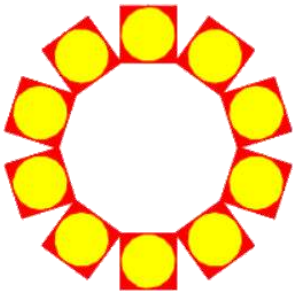
Резултат и програмен код за зад. 7



Фиг. 8

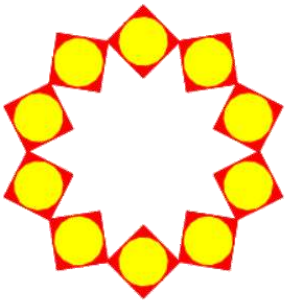
```
import turtle
pen=turtle.Turtle()
pen.shape("triangle")
pen.pensize(3)
pen.speed(5)
pen.color("orange")
colors=["yellow","gold"]
def draw_rombus(side,angle):
  for i in range(2):
    pen.forward(side)
    pen.right(180-angle)
    pen.forward(side)
    pen.right(angle)
a=30
br=16
for i in range(int(br/len(colors))):
  for c in colors:
    pen.fillcolor(c)
    pen.begin_fill()
    pen.left(60)
    draw_rombus(a,60)
    pen.end_fill()
    pen.right(60)
    pen.penup()
    pen.forward(a)
    pen.pendown()
    pen.left(360/br)
```

Задачи за самостоятелна работа. Създайте програми на Python, които изчертават представените композиции от фигури (Фиг. 9, 10 и 11).



Фиг. 9

Фиг. 10



Фиг. 11

Заключение

Представеният подход и набор от задачи способстват за формиране на рефлексивно мислене у учениците в процеса на придобиване на знания и умения в областта на програмирането. Чрез задаване на всяка следваща задача учителят целенасочено поставя учениците в рефлексивна ситуация, при която за разрешаването на новия проблем те използват стари знания и натрупан опит, получен в резултат на вече решени задачи. Този аспект на рефлексивност насърчава учениците да бъдат активни участници в процеса на обучение. Интегрирането на рефлексивни практики в обучението създава предпоставки за повишена ангажираност на учениците и засилва ролята им като активни участници в образователния процес, които осмислят собствените си действия, изграждат умения за критично мислене, оценка и самооценка.

Благодарности

Тази работа е финансирана от проект ФП25-ФМИ-010 „Иновативни интердисциплинарни изследвания по информатика, математика и педагогика на обучението“ към НПД на ПУ.

Литература

[1] Бойкина, Д., Р. Маврова, Рефлексията – движеща сила за развитието на личността на ученика, *Доклади на Юбилейна международна конференция „Синергетика и рефлексия в обучението по математика“*, Бачиново, 2010, стр. 103-109, ISBN: 978 954-423-621-2

[2] Василев, В., Е. Ангелова, Р. Маврова, Комуникативната рефлексия в обучението по математика и информационни технологии, *Доклади на Юбилейна международна конференция „Синергетика и рефлексия в обучението по математика“*, Бачиново, 2010, стр. 495-502, ISBN: 978-954-423-621-2

[3] Тодорова, Е., С. Анева, Т. Терзиева, Формиране на рефлексия в обучението по информатика чрез прилагане на адаптиран модел ALACT, *Доклади на Юбилейна международна конференция „Синергетика и рефлексия в обучението по математика“*, 2020, стр. 311-318, ISBN: 978-619-595-3

[4] Тодорова, Е., *Рефлексията в обучението по информационни технологии в училище*, Университетско издателство „Паисий Хилендарски“, Пловдив, 2024, ISBN: 978-619-202-992-0

[5] УП по КМИТ за 6. клас, https://www.mon.bg/nfs/2023/11/up_vi_kmit.pdf

[6] УП по КМИТ за 7. клас, https://www.mon.bg/nfs/2023/11/up_vii_kmit.pdf

METHODOLOGICAL ASPECTS FOR DEVELOPING REFLECTIVE SKILLS THROUGH THE STUDY OF SCRIPTING LANGUAGES IN MIDDLE SCHOOL

Stefka Aneva[1], Elena Todorova[2,*]

[1, 2] Faculty of Mathematics and Informatics,
Paisii Hilendarski University of Plovdiv, 236 Bulgaria Blvd., Plovdiv

[1] stfaneva@uni-plovdiv.bg

*[2 *] Corresponding author: etodorova@uni-plovdiv.bg*

Abstract. This article presents an approach to teaching scripting text-based languages in lower secondary education, with a targeted focus on fostering reflection within the learning process of the subject “Computer Modeling and Information Technologies”. A set of problems is proposed, along with their specific implementations using the Python programming language, aimed at developing students' algorithmic knowledge and forming reflective skills.