МАТЕМАТИКА И МАТЕМАТИЧЕСКО ОБРАЗОВАНИЕ, 2018
MATHEMATICS AND EDUCATION IN MATHEMATICS, 2018
Proceedings of the Forty-seventh Spring Conference
of the Union of Bulgarian Mathematicians
Borovets, April 2–6, 2018

ОБУЧЕНИЕ ПО ПРОГРАМИРАНЕ В РАННА ДЕТСКА ВЪЗРАСТ*

Красимир Манев, Адриана Панайотова, Весела Симова, Поли Янкова, Ивайло Бонев

Програмирането на компютри е една от най-модерните и перспективни професии на нашето време. За да се осигурят достатъчно кадри за софтуерната индустрия, не само в национален, но и в световен мащаб, е необходима систематична и всеобхватна подготовка на тесните специалисти в областта. Световна тенденция е обучението по програмиране да започва в по-ранна възраст. България не трябва да изостава от тази тенденция, ако иска да има развита софтуерна индустрия и модерна икономика. В тази статия се разглеждат някои проекти за ранно обучение на деца по програмиране, реализирани у нас, и перспективите за разширяване на обхвата на тази дейност.

1. Въведение. Не подлежи на дискусия фактът, че *Информатиката* и *Информационните технологии* са ключови области в съвременното общество. Освен за решаване на многобройни задачи от областта на науката и техниката, съвременните компютри и компютризирани системи (мобилни телефони, таблети, роботи, други устройства с вградени управляващи компютърни модули и т.н.) намират все по-широко място в живота на човека – в пряката му професионална дейност, за да бъде своевременно информиран за случващото се в природата и обществото, както и да организира и прекарва приятно свободното си време. Всичко това се дължи на необятните възможности на компютърната техника, които позволяват едно и също устройство да бъде използвано по най-различен начин и за подпомагане на най-различни дейности. И всичко това, благодарение на гениалното хрумване на учения с български корени Джон Атанасов, постулирал необходимостта съвременните компютри да разполагат с *регенеративна оперативна памет* – такава, че в нея да могат да се зареждат за изпълняване **различни програми**. Затова *компютърното програмиране* е дейност с огромно обществено значение, а професията *програмист* (или както е модерно да се казва напоследък – *софтуерен инженер*) – една от най-перспективните.

В самото начало на навлизането на компютрите в живота на човека, програми се пишат само от научните работници. Не съществуват технологични инструменти

* **ACM Classification Codes:** K.3.2: Computer and Information Science Education
Тази работа е частично подпомогната от Сдружение „Европейска юношеска олимпиада по информатика“.

за облекчаване труда на програмистите – програмите се пишат с прости текстови редактори на *машинните езици* на съответните компютри, или на създадените за улеснение *асемблерни езици* – много близки до машинните. Постепенно кръгът от програмисти се разширява и се създават *езици за програмиране*, ориентирани към определени класове задачи – FORTRAN (предназначен за програмиране на числените методи на математиката и математическата физика), COBOL (за програмиране на задачите на икономиката и бизнеса) и др. Естествено се появяват и *универсални* езици за програмиране, като „тежкия“ и бързо изчерпал възможностите си PL/I и много по-гъвкавия и станал основа на много други езици C. Появяват се и първите *среди за програмиране*, обединяващи в себе си всички технологични инструменти, необходими за създаването на един програмен продукт: текстов редактор – за подготовка на текста на програмата, транслатор – за превод на програмата до машинен език или за интерпретиране на кода, свързващ редактор – за оформяне на програмата като изпълнима, дебъгер – за откриване и отстраняване на логическите грешки, системи за поддържане на различни версии и т.н. За редица типични подзадачи се създават подпрограми. От дейност, **близка до изкуството**, създаването на програми се превръща в **производство**.

С появата на *графичните монитори* се увеличават областите, за които се създават компютърни програми. *Графичният интерфейс* прави използването на програмите лесно за много широк кръг потребители. Широкото навлизане на интернет води до появата на *мрежовото програмиране* и все по-нови, неподозирани доскоро възможности, за използване на компютърни програми. Вграждането на компютърни устройства в роботи, автомобили, домашни уреди и т.н. днес е повсеместно. Каквито и промени да настъпят в бъдеще, създаването на нови компютърни програми ще бъде една от основните области, която ще предлага възможност за работа.

Значението на програмирането, като вид човешка дейност, е в процес на всемирно осъзнаване. В бъдеще, незнанието на поне един език (и съответна среда) за програмиране ще се счита за неграмотност, а съвсем скоро 95% от професиите ще изискват дигитални умения. За съжаление, в световен мащаб, образованието все още не може да отговори на това предизвикателство. Въпреки че децата вече бързо се научават да си служат с новите технологии, дори ненавършили 7 години, знания и умения по програмиране те получават, в най-добрия случай, едва в гимназиалния курс и то в недостатъчна степен. Единици са страните като Великобритания, Естония и Финландия, които са въвели *Информатиката* като предмет още в първи клас и това ще им даде безусловно предимство в обозримо бъдеще.

За да бъде тази област на човешката дейност развивана, са необходими доста кадри. В момента софтуерната индустрия е в недостиг на качествени програмисти и други специалисти (бизнес анализатори, тестери, дизайнери и др.), необходими за производството на софтуер. За да бъде запълнен недостигът от програмисти, е необходима добре развита система за тяхната подготовка. Още повече, че по-богатите държави с лекота „изсмукват“ голям брой от подготвяните по света кадри.

Какво е положението у нас в момента. За малки и не много богати страни като България, но с много силно развита софтуерна индустрия, нуждата от програмисти е особено силна. В средното ни училище ударението е поставено върху изучаването на *Информационни технологии* – дисциплина, която е призвана да даде на обучаваните основни знания и умения **да си служат с компютър** в ежедневната си

дейност, но не и да подготвя бъдещи програмисти. ИТ се изучава като свободно избираема учебна дисциплина в началния курс, и като задължителна от 5. до 10. клас. За сметка на това *Информатиката* се преподава в рамките на една година в регулярното училище и с повече часове само в някои математически гимназии. В общия случай, това преподаване е със съмнително качество. Така се подхранва вредният стереотип, че програмирането е дисциплина, достъпна само за талантливи ученици. Затова и много от учениците, постъпващи в университетските специалности по *Информатика* не са готови за обучението, което ги очаква там.

Университетите ни, които имат специалности, свързани с *Информатиката* и *Информационните технологии*, включително и в областта на хардуера, не са малко. Но броят на постъпващите там студенти не е голям, а в резултат на сравнително нелекото обучение, **броят на завършващите е недостатъчен**, за да задоволи дори нуждите на нашата софтуерна индустрия. Друг проблем на университетите е много ниското равнище на заплатите на преподавателите, които са в пъти по-ниски от тези в софтуерната индустрия. Затова университетите страдат от **хронична нужда от преподаватели** и това не може да не се отрази на равнището на преподаване.

Всеобщо е мнението на всички колеги в областта, че това не може да продължава повече така и са нужни сериозни усилия – и от страна на колегията, и от страна на обществото и държавните органи – за значителни промени в обучението на информатици. Тази статия е посветена на една от съвременните тенденции в областта на подготовката на програмисти – обучението по програмиране на деца в ранна детска възраст. Дълбокото убеждение на авторите е, че такова обучение е не просто възможно, а задължително. Така, както изучаването на фундаменталните дисциплини *Роден език* и *Математика* започва в най-ранна детска възраст и продължава през целия период на обучение, така трябва да бъде и с дисциплината *Информатика*, която вече е фундаментална.

2. Програмиране и компютърно програмиране. Тази статия е посветена основно на *компютърното програмиране*. *Програмирането* обаче е много по-стара човешка дейност и възниква далеч преди създаването на компютрите. С много голяма вероятност може да се твърди, че първото *устройство с програмно управление* е създадено през 1805 г. от френския текстилен инженер Жан-Мари Жакард. Работата на Жакардовия тъкачен стан се управлява от предварително подготвена перфорирана хартиена лента (*програма*), в която са кодирани управляващи инструкции за стана така, че върху тъканта да се получават различни фигури. Със **смяна на лентата**, станът е готов да тъче тъкан с други фигури, според визията на човека, който подготвя лентата.

Стана на Ж.-М. Жакард

На подобен принцип са построени и различни музикални инструменти (*латерни* и други подобни устройства), които **автоматично**, без намесата на човек изсвирват мелодия, която е запрограмирана с подходящо поставени върху метален барабан зъбци. При въртенето на барабана всеки зъбец докосва метална пластина, която

звучи със съответен тон. Със смяна на барабана, латерната започва да свири нова мелодия.

Но това не са единствените човешки дейности, които можем да наречем програмиране. Най-общо казано, *програма* е всяка последователност от добре дефинирани примитиви – неголям брой елементи или операции, предназначена за създаване на някакъв обект или постигане на определен ефект. Така всеки проект за построяване на една сграда можем да наречем програма, тъй като предписва на строителите строга последователност от познати за тях операции, които трябва да изпълнят, за да построят сградата. По същия начин програма е и всяка последователност от операции за изработване на машинен детайл или от инструкции за сглобяването на сложна машина или уред. Програма можем да наречем и всяка рецепта за приготвяне на определено ястие, тъй като достатъчно точно описва нужните за целта продукти и стъпките, които изпълняващият рецептата трябва да извърши.

В обучението по елементарна математика съществуваше типична програмистка дейност, която е абсолютно незаслужено пренебрегвана в наши дни – построяването на зададена геометрична конструкция с помощта на елементарни действия с линийка (за чертаене на линии) и пергел (за чертаене на окръжности). Програма е и всяка процедура, която работещите в корпоративната и публичната администрация са задължени от нормативните документи да изпълняват в ежедневната си работа, обслужваща дейността на фирмата или нуждите на гражданите. Програмиране може да открием и в многобройните детски играчки-конструктори, при които детето трябва да изгради зададен обект от примитивни елементи – кубчета или други обемни елементи. Със **създаване и изпълняване на програми** са свързани още много човешки дейности.

С построяването на първите компютри дейността, наричана програмиране, все по-тясно се свързва със създаването на програми за тези сложни устройства – **компютърното програмиране**. Фактически това, което компютърът може да прави, е да изпълнява програми, съставени от набор от стотина примитивни инструкции – като аритметичните операции и сравняването на числа, както и да сменя реда на изпълняване на инструкциите в зависимост от получените от операциите резултати. Така, с подходящо комбиниране в програма на някои от тези не много примитиви, компютърът е в състояние да извършва огромно количество и най-разнообразни дейности, необходими на човека.

В сравнение с всяка друга дейност, която може да наречем програмиране, компютърното програмиране е с невероятно **по-големи възможности** и постепенно компютърните програми изместиха много от класическите форми на програмно управление – механични или електронни, като доведоха и до значителен прогрес в редица други човешки дейности, в които не бяха използвани програми и програмно управление. Очакванията са, за в бъдеще този процес на използване на компютърни програми в практиката да бъде доминиращ. Защото възможностите на компютърните програми са огромни и понякога неподозирани.

Освен своя огромен **потенциал и невероятна ефективност**, компютърното програмиране има и едно друго много голямо предимство – то е **изключително евтино**. Компютър, използващ незначително количество електроенергия, среда за програмиране и подготвен програмист, са достатъчни за създаването на продукт с много висока цена за сравнително кратко време. Затова и обучението в компютърно

програмиране е изключително евтино, защото не използва скъпи механични или електронни елементи. В този смисъл, най-евтиният и най-ефективен начин да се запознаят подрастващите с принципите на програмното управление, с начините за изграждане и изпълняване на произволни програми е, да се обучават да създават компютърни програми.

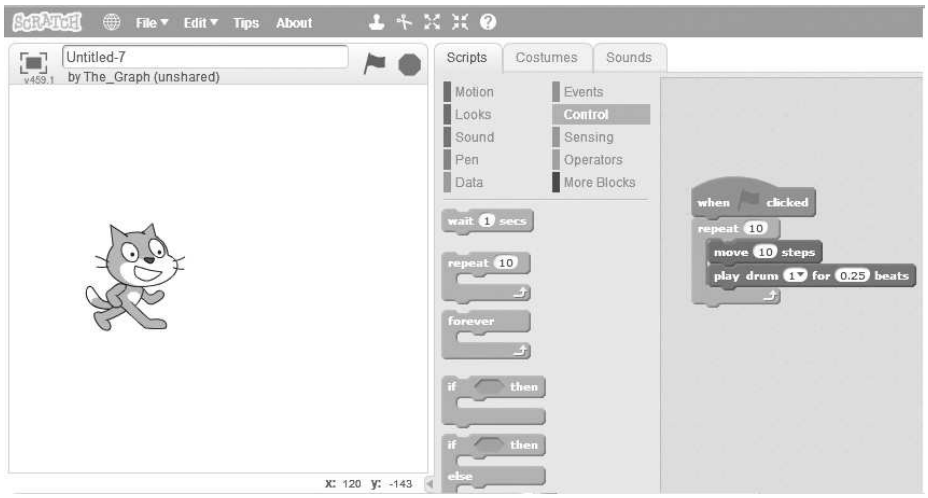
3. Програмиране в ранна детска възраст. Обединяващата концепция на авторите на тази статия е, че обучението по програмиране трябва да бъде **систематично** и **всеобхватно**. Систематично, защото само така може да се постави една сериозна основа за изграждане на достатъчно много специалисти, способни да се справят с предизвикателствата на времето и да създават необходимия за обществото и отделните личности софтуер. И всеобхватно, защото няма област на човешката дейност, която да не изисква, днес и/или в бъдеще, създаването и използването на компютърни програми. Няма друг начин да се осъществи тази концепция, освен с въвличането в обучение по програмиране на все повече обучаеми – от най-ранна детска възраст до последните курсове на университетските програми, а и след това, на работното място на всеки индивид.

Осъществяването на концепцията няма да е лесно и ще изисква време. Необходими са дълбоки промени, за да се достигне до разбирането за необходимостта от изучаване на програмиране, за да се изработят съответни държавни стратегии и да се претворят тези стратегии в реалност. И първата стъпка в тази посока е, да се започне с обучение по програмиране на повече деца в ранна възраст.

Идеята за обучение по програмиране на деца не е нова. Една от първите среди за програмиране с явна насоченост към най-малките е езикът за програмиране Logo, създаден през 1967 година от Seymour Papert и сътрудници във фирмата за изследване и развитие в областта на информатиката Bolt, Beranek and Newman (BBN), в Cambridge, щата Massachusetts, САЩ. Разработката е осъществена в тясно сътрудничество с такива корифеи на теоретичната информатика и разработването на интелигентен софтуер като Marvin Minsky и John McCarthy от Масачузетския Технологичен Институт (MIT). Езикът Logo [1] е от функционален тип (надстройка на Lisp) и, макар че по същество е универсален, основната му цел – да бъде интересен за деца – е осъществена посредством ориентацията му към създаване на графични изображения, чрез управляван от програмите рисуващ обект – Logo-костенурката. Така се преодолява основна трудност при обучението по програмиране на деца – сравнително късното достигане до създаване на програми с ефектен резултат, типично за езиците за програмиране, предназначени за създаване на сериозни програмни продукти. Езикът Logo беше много популярен у нас в рамките на проекта ПГО [2].

Причината Logo да не получи широко разпространение, освен неосъзнатата от обществото необходимост от ранно обучение по програмиране, според нас, е в това че езикът е функционален[1]. Докато в широката програмистка практика се използват предимно процедурни езици. Затова естествена стъпка в развитие на идеята е била, да се създаде език и среда за програмиране, която да има атрактивността на Logo, но да е в процедурен стил. Така през 2003 г. се появява езикът за програ-

[1] Всъщност Лого може да се използва и за функционално, и за процедурно програмиране (бел. ред.).



Среда за програмиране на Scratch

миране Scratch, разработен в лабораторията Media Lab на MIT от групата Lifelong Kindergarten, ръководена от Mitchel Resnick, в сътрудничество с канадската консултантска фирма Playful Invention Company [3]. Докато Logo е надстройка на Lisp, то за Scratch може да се каже, че е надстройка на популярния език C, оказал огромно влияние върху съвременното програмиране. Основната идея, както и при Logo, да дава възможност бързо да се напишат ефектни програми, за да бъде поддържан интересът на обучаемите, е осъществена с ориентацията му към създаване на обекти, добре познати и харесвани от най-малките – анимационни клипове със съпътстващо озвучаване.

Средата за програмиране на Scratch е визуална, основните процедурни примитиви и вградените стандартни процедури са оформени като елементи на пъзел, които може да се редят един до друг или да се вграждат един в друг за постигане на поставената цел (*drag&drop програмиране*). Повечето от елементите лесно се настройват, като имат поле, в което да се постави настройващият параметър. На разположение на обучавания са множество анимационни герои, които той може да движи по работното поле според замисления от него сценарий. Синтаксисът на Scratch за сглобяване на програмата от пъзелни блокчета оказва огромно влияние при разработване на нови езици за обучение на деца и е считан за de facto стандарт за разработване на такива езици.

Като пример на най-модерен език, който може да бъде използван за обучение на деца, може да посочим езика Python [4]. Като представител на господстващата днес програмистка парадигма – обектно-ориентираното програмиране, той обединява много от фундаменталните програмистки концепции, необходими за създаване на комерсиален софтуер. Но едновременно с това съдържа и редица елементи, правещи го много подходящ за обучение на деца. На първо място това е позабравената концепция за директно интерпретиране на операциите и операторите, която дава възможност децата да създават ефектни, и задържащи вниманието, приложения на много ранен етап от обучението. Езикът Python предлага още възможност за изпол-

зване на познатата ни „костенуркова графика“ на Logo, а също и удобни, и лесни за използване средства за създаване на анимационни клипове и игри, както в Scratch.

В наши дни инструментите за обучение по програмиране на деца от най-ранна детска възраст, включително и предучилищна, са много. Създаването им ще продължи и в близко бъдеще, ще имаме на разположение още по-добри такива инструменти. Като финал на този раздел бихме искали да цитираме думите на създателя на Scratch Mitchel Resnic, казани за създадения от него език, но важащи за всички подобни среди за програмиране – думи, които напълно се покриват с нашето виждане по темата: „*Способността да създаваш компютърни програми е важен елемент на грамотността в съвременното общество. Когато хората се учат да програмират на Scratch, те научават важни стратегии за решаване на проблеми, начини за реализация на проекти и идеи за комуникация.*“ [5].

4. Обучение по програмиране за деца у нас. Съвсем естествено, България – страна с големи традиции в обучението по програмиране на деца в различни възрасти и с достатъчно добре развита софтуерна индустрия – не би могла да остане встрани от една очертаваща се тенденция. През последните години у нас, главно благодарение на инициативата на различни фирми, свързани със софтуерната индустрия, започна активното обучение по програмиране на деца в ранна възраст. В този раздел представяме накратко програмите на някои от тях.

4.1. Програма на Обучителен център „Технократи“. Обучителен център „Технократи“ работи с деца на възраст от 7 до 18 години, като се фокусира върху две основни теми: *Роботика и програмиране* и *Възобновяеми енергийни източници*. Тъй като тази статия е посветена на обучението в ранна детска възраст, ще се спрем тук само на работата по първата тема. Курсовете по *Роботика и програмиране* включват няколко етапа: конструиране на робот; запознаване със сензори и механика; програмиране на робот; състезания и демонстриране на наученото.

Това, което беше много интересно и изненадващо в началото, е, че програмирането се оказва по-интересно за децата от конструирането на самия робот. Разбира се, това много зависи и от индивидуалността на детето, но в общия случай програмирането много интригува децата. Фактът, че манипулират с робота чрез компютъра, за тях е истинско предизвикателство. Колкото и да звучи банално, програмирането на роботи определено разкрива нов хоризонт пред децата и буди техния интерес.

Възрастовата група за обучение по *Роботика и програмиране* включва деца на възраст от 7 до 14 години, разделени на две групи – 7–10 и 10–14 години. Важен подход в това обучение е, да няма по-голямо разделение на възрастов принцип, а да се работи в групи от деца с различна възраст, с естествена групова динамика и възможност едни деца да се учат от опита на други. Много силно впечатление прави страстта, с която по-знаещите се учат от по-малко знаещите, и с която по-знаещите обясняват на по-малко знаещите процеса на конструиране и програмиране на робота. При това, по-знаещи не означава непременно по-големи! Опитите да се представи конструирането и програмирането пред деца в предучилищна възраст (3–6 години) показва, че и там има интерес, но не е лесно да се задържи вниманието и е трудно трайно да се надградят знания, които да бъдат използвани в бъдеще. Ако направим аналогия с училището, в тези групи процесът повече прилича на случващото се в „занималня“, отколкото на случващото се в „клас“.

Важни елементи на програмата са:

• събиране на деца в различна възраст в група и работа в екип по повечето от задачите;

• обучаващите не са „учители“, а „ментори“, които насочват децата и ги подтикват да задават правилните въпроси и да участват в дискусии. Отношението на менторите към децата е като към възрастни, а децата се учат от менторите, така, както и менторите се учат от децата;

• дискусии се провеждат по всеки възникнал въпрос – никой въпрос не се счита за глупав или безсмислен и никой въпрос не остава без отговор;

• поддържа се непрекъсната връзка с родителите;

• организират се състезания, в които ударението е поставено повече на партньорството, отколкото на противопоставянето.

Поради натоварения график на децата през седмицата, а и през почивните дни, повече от 1–2 занятия седмично биха били проблем (целта е да се събуди интерес, а не да се постигне нежелание). Опитът показва, че 2 занятия седмично по 2 часа е оптималната натовареност. Затова се провеждат по 4–8 модулни занятия месечно, с общ обем от 8-16 часа, в които основно се набляга на програмирането. Освен това се предлагат уикенд-лагери, които децата възприемат като време за игри и забавления и в тях стремежът е да се представи програмирането като забавление. Допълнителна форма на обучение са нерегулярните безплатни занятия в училищата, в които учат децата (в София и в страната) или в офиса на фирмата, целта на които е популяризирането на роботиката и програмирането.

В обучението по *Роботика и програмиране* използваме две платформи. Платформата ModKit [6] е разработена по модела на Scratch (drag&drop) и работата с нея – редене на цветни плочки за различните операции – е лека и забавна. Тя бързо скъсява дистанцията между ментора и децата, и им представя процеса на програмиране като игра. Платформата RobotC [7] е доста по-сложна. С нея се надгражда наученото в предишните модули и, след като са овладели основите на програмната логика, децата се въвеждат в света на истинското програмиране. Тя също предоставя възможност за drag&drop програмиране, но има възможност и за директно писане на програмен код. Децата се обучават да работят с нея, само след като са преминали началните модули с ModKit и са показали качества и интерес да се занимават с програмиране.

За да се включи дете в програмата **не е нужна предварителна подготовка**. Предизвикателство е да представим на децата нещо, за което не са били предварително подготвени и да проследим как се справят те, а и как се справят менторите. За целта разчитаме най-вече на въвеждането на новите елементи под формата на игра. **Групите са малки** – 12–14 деца – за да се работи пълноценно с всяко дете и да се поддържа динамиката на групата. Важно е всяко дете да почувства, че му се отделя нужното внимание. А също така по-знаещите да „изтеглят“ останалите напред, а не обратното. По отношение на **проверката на успеваемост**, концепцията е, че традиционното поставяне на „оценки“ не е подходящо. Затова се изисква обучаваните да демонстрират наученото , като го приложат за направата на нещо работещо – например, робот, изпълняващ зададени функции.

4.2. Програма на MindHub. Обучението в MindHub се провежда по собствена учебна програма, изцяло проектно ориентирана, която позволява на деца да

започнат обучение на 6 години и да завършат обучението с изучаване на език за програмиране на 10 години. Методите и инструментите са иновативни и разработени специално за обучение по програмиране на деца, изцяло съобразени с възрастта на обучаваните и интегрирани в процеса на обучение по интересен начин. Програмата е разделена на три сегмента, в зависимост от възрастта на която детето започва курса на обучение – MindHub Junior (за деца на възраст 6–7 години), MindHub Middle (за деца на възраст 8–9 години) и MindHub Senior (10 и 11-годишна възраст). Всеки от модулите е с продължителност 4 седмици, с по 2 часа седмично.

Целта на сегмента MindHub Junior е да развие уменията на децата за аналитично и логическо мислене и основни знания за процеса на програмиране, като за целта се използват разнообразни игрови методи. Обучението преминава през 21 модула за основи на програмирането и 9 модула за програмен език, с плавен преход от решаване на логически задачи и казуси, през изучаване на основите на програмирането, за да достигне в края до изучаване на език за програмиране (Python). Системата на обучение е ориентирана към изграждане на подходяща социална среда. Работи се в малки екипи (8–10 ученици), в които децата да развиват и своите умения за общуване и за представяне на извършеното.

Сегментът MindHub Middle цели да доразвие у децата уменията за аналитично и логическо мислене, както и да изгради умения за разрешаване на проблеми, дигитални умения и компетенции, които да ги подготвят за бъдещото им развитие. В този сегмент се залага основно на изучаване на компютърното програмиране (17 и 9 модула, съответно) под формата на проектно-ориентирано обучение, с плавно преминаване в последните модули към изучаването на езика за програмиране Python 3 (IDLE). След запознаването с този език децата вече са в състояние да създават достатъчно сложни приложения – игри, уебсайтове и др. И в този сегмент системата е ориентирана към развитие на уменията за общуване и представяне на извършеното.

Сегментът MindHub Senior не се различава съществено от сегмента MindHub Middle. Тук процесът на обучение е много по-интензивен, съставен от 18 кратки модула програмиране и най-вече работа по проекти. Езикът за програмиране е също Python.

Среда за drag&drop програмиране

Като среда за програмиране на Python се използва най-вече IDLE [8], като за обучение по програмиране се използват също Wonder, Blockly, Scratch, Scratch JR и др.

От методическа гледна точка обучението залага на индивидуалния подход към всеки ученик, към насърчаване на неговата креативност и развиване на комплексни умения за постигане на успех в работата. Създава се приятелска социална среда, работейки в малки групи, с деца на приблизително еднаква възраст. Полагат се усилия за ангажиране на вниманието на всяко дете през цялото времетраене на занятията. При оценката на постигнатото се залага на двукомпонентна

обратна връзка – оценяват се както проектите, създадени от децата, така и работата на менторите, по определени критерии за резултатите от работата им.

4.3. Програма на Coder Dojo. Ученето на факти в свят, в който Google може да отговори на почти всеки въпрос, е все по-маловажно. Нужно е да се развиват умения, които ще дадат на децата възможност да успяват в утрешния ден. И едно от тези умения е умението да се програмират компютри и компютризирани устройства. Участниците в движението Coder Dojo вярват, че умението да се използва език за програмиране е все по-важно в съвременния свят. По-добре и по-лесно е тези умения да се усвоят в максимално ранна възраст и никое дете не трябва да бъде лишавано от тази възможност.

Coder Dojo изгражда глобална мрежа от локални клубове за обучение по програмиране на деца. Те се ръководят от доброволци. Всяко дете на възраст от 7 до 17 години, може да посещава Dojo клуб, където да се научи да програмира и да опознае технологиите в неформална творческа и социална среда. Обучението е ориентирано към разработване на проекти в екип с ментор. Менторите на Coder Dojo са високо квалифицирани специалисти в областта на *информатиката* и *информационните технологии*. Благодарение на тях, Coder Dojo успява да осигури безплатно обучение по програмиране за деца и така да запълни донякъде липсата в традиционното образование.

Чрез своя менторски модел на обучение програмата Code Dojo се стреми да постигне:

• развитие на умения, от които всяко дете ще се нуждае в бъдеще – компютърно мислене, трансдисциплинарна грамотност, способност да формулира задачи и планира работни процеси за тяхното реализиране, адаптивно мислене и социална интелигентност;

• усвояване на технологични умения от най-ранна детска възраст;

• увеличаване броя на момичетата с интерес към *информационните технологии*.

Основната учебна програма включва запознаване с базови понятия, като алгоритъм, програма, цикъл, условие, променлива и т.н. Чрез използване на интерактивни и забавни похвати се осигурява бързото овладяване от децата, развиване и свободно боравене с основните програмистки умения. Същевременно децата, обучаващи се в Coder Dojo, се насърчават да работят в екип, да търсят нужната им информация, да структурират идеите си, да ги реализират и да ги представят пред публика. По време на обучението децата се запознават с: езици за програмиране като ScratchJr, Scratch и Java Script в подходящи за деца среди за drag&drop програмиране; езици за създаване на Web-сайтове като HTML и CSS; специализиран хардуер като micro:bit и Makey Makey; и др.

Моделът на Coder Dojo е доста успешен, защото се реализира от доброволци, които вярват в необходимостта от ранно обучение по програмиране и изпитват нужда да са част от общност на съмишленици. Това създава неформална среда, основана на иновации, приобщаване и творчество, участниците в която уважават и ценят идеите на другите, техните вярвания и култура.

5. Перспективи. Както се вижда от казаното по-горе, възможностите за обучение по програмиране на деца в ранна възраст са големи. А и опит у нас за провеждане на такова обучение не липсва. Ясно е също така, че възможностите на фирмите,

които са се заели с реализацията на идеята, не са неограничени. Това, което липсва в момента, е всеобхватността. Необходимо е обучението по програмиране на деца да стане част от редовния учебен процес, като за целта се намерят подходящи форми.

Извънкласните школи са една от възможностите. Националният екип за олимпиади по информатика към Съюза на математиците в България, близо 40 години вече, организира такова обучение за неголеми групи от много мотивирани ученици, които се подготвят за участие в Националната олимпиада по програмиране и в Международните олимпиади [9]. Най-малките участници в олимпиадите, обаче, са на възраст 10–11 години, тъй като в по-ранна възраст олимпийската подготовка не е възможна. Естествено ще бъде, подобни като организация школи, но за по-малки ученици да се организират в училищата. Трудността тук ще бъде да се намерят достатъчно мотивирани и подготвени учители, които да се заемат с ръководството на такива школи, като използват опита на някоя от споменатите по-горе фирми, както и да се осигури финансирането на такива школи.

Друга подходяща форма е **разработването на проекти** в областта на информатиката. Подобна форма на обучение също се осъществява от СМБ, в рамките на Ученическия институт по математика и информатика [10], но в нея са включени предимно ученици от гимназиалната степен. От казаното по-горе се вижда, че проектно-ориентираният подход за обучение по програмиране е не само възможен, но и практикуван в някои от фирмените програми. И тази идея за разширяване на обучението по програмиране, обаче, не може да бъде осъществена без привличането на повече учители и доброволци от софтуерния бизнес.

И все пак, за да бъде постигната така желаната всеобхватност на процеса, най-добре е **обучението по програмиране да бъде инкорпорирано, трайно и систематично, в учебните програми на българското училище** – от първата до последната година. Тази идея е представена отдавна в [11]. За целта е крайно време в българското училище да бъде въведен предметът *Информатика и информационни технологии*, в който по естествен начин да бъдат обединени двата изкуствено разделени предмета. Такава стъпка би била в пълен сихрон със съществуването на други „интегрални“ предмети, като *История и цивилизация*, *Биология и здравно образование*, *Химия и опазване на околната среда*. Само така може да се постигне необходимата сплав между фундаменталните знания по информатика и използването им за решаване на реални задачи.

6. Заключение. Няма съмнение, че професията компютърен програмист е една от най-перспективните в наши дни и нуждите на пазара на труда за кадри с такава професионална квалификация са големи и непрекъснато нарастват. Освен това, в много съвременни професии се срещат дейности, които имат характера на програмиране и извършването им е невъзможно без подходяща подготовка. Няма съмнение, също така, че обучението по компютърно програмиране е най-евтиният и ефикасен начин за изграждане на програмистки умения, независимо от вида на средствата за програмиране. Затова обучението по програмиране трябва да бъде систематично и всеобхватно.

България е една от страните с най-голям опит в обучение по програмиране в ученическа възраст. И в академичните среди, и във фирмите, свързани с производство на софтуер, е натрупан безценен опит, който може да бъде използван за постигане на

необходимата систематичност и всеобхватност. Нужни са само приоритетна визия за бъдещето на професията в обществото и държавните институции за постигане на тази цел.

Световна тенденция е обучението по програмиране да започва в ранна детска възраст и тази тенденция отдавна е намерила почва у нас. В тази статия показахме някои от възможностите за такова обучение. Добре би било тези възможности да се използват, като се популяризира опитът на компетентните организации и се разшири кръгът на обучаемите. А затова не е необходимо да се изчаква обществените и държавните органи да се ангажират с проблема, тъй като всяко забавяне ще има лоши последици върху българския софтуерен бизнес, който е измежду най-добре развиващите се и печеливши отрасли на българската икономика.

ЛИТЕРАТУРА

[1] S. Papert. What is Logo? Who needs it? In Logo Philosophy and Implementation, LCSI (1999), V—XVI, http://www.microworlds.com/support/logo-philosophy-implementation.html (5.01.2017)

[2] Е. Сендова. Конструкционизмът като образователна философия и култура в български контекст – в памет на Сиймър Пепърт. *Математика и математическо образование*, **46** (2017), 29–54.

[3] M. Resnick et al., Scratch: Programming for All. *Communications of the ACM*, **52**, No 11 (2009), 60–67. doi:10.1145/1592761.1592779

[4] J. R. Briggs. Python for Kids, https://doc.lagout.org/programmation/python/Python%20for%20Kids_%20A%20Playful%20Introduction%20to%20Programming%20[Briggs%202012-12-22].pdf

[5] M. Resnick. https://scratch.mit.edu/about/.

[6] MODKIT. Program Your World. https://www.modkit.com.

[7] ROBOTC: a C Programming Language for Robotics, http://www.robotc.net/.

[8] 25.5 IDLE, https://docs.python.org/3/library/idle.html.

[9] K. Manev et al. Programming Contests for School Students in Bulgaria. *Olympiads in Informatics*, **1** (2007), 112–123. http://ioinformatics.org/oi/shtm/INFOL010.shtml

[10] Ученически институт по математика и информатика, http://www.math.bas.bg/omi/hssimi/

[11] К. Манев. Информатика и информационни технологии в българското училище. *Образование и технологии*, **1** (2010), 42–47, ISSN 1314-1791

Красимир Манев
Департамент Информатика, НБУ
бул. „Монтевидео“ № 20
1164 София, България
e-mail: kmanev@nbu.bg

Адриана Панайотова
e-mail: adi@coderdojo.bg
Весела Симова
e-mail: vessi@coderdojo.bg
Coder Dojo България
бул. „България“ № 111
София, България

Поли Янкова
MindHub
ул. „Солун“ № 62
1680 София, България
e-mail: polly@mindhub.bg

Ивайло Бонев
Обучителен център „Технократи“
ул. „20 април“ № 13
1606 София, България
e-mail: ivaylo@technokrati.bg

TEACHING PROGRAMMING FOR CHILDREN

Krassimir Manev, Adriana Panayotova, Vesela Simova, Poly Yankova, Ivaylo Bonev

Computer programming is one of the most modern and promising profession of our days. For preparing enough specialists for the software industry, not only for the country but worldwide, a systematic and comprehensive system for education is necessary. A world trend is the education in programming to start as early as possible. Bulgaria should not deviate from this trend if it would like to develop software industry and modern economy. The paper presents the concept of early education in programming of children implemented in Bulgaria and the perspectives for extending the scope of such activities.