



ЧПГ „АК-АРКУС“ ЕООД
cognitio libertas perfectio



ecis

ETS AUTHORIZED
TOEFL iBT CENTER

Утвърждавам:.....

Петър Иванов – директор на
ЧПГ „АК-Аркус“ ЕООД



**УЧЕБНА ПРОГРАМА
ПО ИНФОРМАТИКА
ЗА IX КЛАС
ОБЩООБРАЗОВАТЕЛНА ПОДГОТОВКА,
ПРОФИЛ ЧУЖДИ ЕЗИЦИ**

Настоящата учебна програма съответства напълно на изискванията на ЗПУО, на Наредба №5 от 30 ноември 2015 г. за общообразователна подготовка и на Наредба №7 от 11 август 2016 г. за профилирана подготовка. В нея са интегрирани методически препоръки от Middle Years Programme (MYP) – междинната степен на Програмата на Международния Бакалауреат. Програмата е разработка на специалисти от ЧПГ „АК-Аркус“ ЕООД, гр. Велико Търново и е част от документацията, свързана с кандидатстването на гимназията за иновативно училище през 2018 г.

Адрес:

ул. Драгоман 16, гр. Велико Търново, 5000, България, Телефон: +35962619959, Факс: +35962619961

УЧЕБНА ПРОГРАМА ПО ИНФОРМАТИКА ЗА IX КЛАС

ОБЩООРАЗОВАТЕЛНА ПОДГОТОВКА

Кратко представяне на учебната програма:

Информатиката е наука, която се занимава с методите за структуриране, събиране, обработка и разпространение на данни. Постиганията на тази научна област са пряко и динамично свързани със създаването и развитието на високотехнологични средства (компютри, операционни и комуникационни системи, потребителски софтуер, софтуер за разработка на приложения и др.), които са важен инструмент и/или инфраструктура, върху която функционират почти всички сфери на съвременното общество. В тази връзка учебната програма (а и обучението като цяло) по информатиката в IX клас е съобразена със система от взаимосвързани фактори, по-важните от които са:

1. Мястото на предмета в учебния план:

- брой на часовете;
- връзките и значението му (в хоризонтален и вертикален план) с другите учебни предмети, особено с математика и информационни технологии (ИТ).

2. Съвременното състояние, понятийният апарат, логическата структура, методите и средствата на научната област.

3. Възрастовите (познавателни и психологични) възможности на учениците.

4. Процесът на обучението по информатика притежава потенциални възможности за личностно изграждане и развитие на ученика – формиране на абстрактно и логическо мислене, възпитание и формиране на адекватно отношение към заобикалящата действителност.

5. Учебният предмет е и възлов инструмент, с помощта на който могат да се мотивират, ориентират, привличат и развият младите хора, така че впоследствие (по-горните класове и университетите) да бъдат

създадени специалисти, способни да осигурят кадрово информационното обслужване на всички обществени сфери.

ОЧАКВАНИ РЕЗУЛТАТИ В КРАЯ НА КЛАСА:

Области на компетентности	Знания, умения и компетентности <i>В резултат на обучението си ученикът</i>
Информатика	- Описва предмета и ролята на информатиката за моделиране - Познава представянето на информация във вид на данни - Посочва примери на обекти и явления, при които е практически приложимо използването на средствата на обектно-ориентираното моделиране
Обектно-ориентирано Програмиране	- Обяснява основните етапи при създаване и изпълнение на компютърна програма - Прилага обектно-ориентиран подход при създаване на несложна компютърна програма - Описва основни начини за създаване, изпълнение и тестване на програмен проект в интегрирана среда за разработка с използване на визуални графични средства - Използва библиотеки от готови компоненти - Спазва добър стил на програмиране
Графичен потребителски интерфейс	- Разбира и използва основни компоненти на среда за визуално програмиране при разработка на софтуер - Проектира графичен потребителски интерфейс с визуални средства - Избира подходяща графична компонента в съответствие с необходимата функционалност на графичния интерфейс - Умее да настройва свойствата на графичните компоненти - Програмира подразбиращи се събития за основни компоненти от графичния интерфейс
Алгоритми и структури от данни	- Разбира същността на „тип данни“ - Разграничава различни типове данни - Разбира същността на алгоритмите и начини за описанието им - Определя подходящ тип данни за определена задача - Прилага основни управляващи конструкции - Прилага основни алгоритми за намиране на сума, минимален/максимален елемент и средно аритметично
Софтуерни	- Използва визуално програмиране за решаване на несложни задачи

приложения	- Създава програми за графично изобразяване на геометрични обекти със стандартни средства в езика - Създава програмни приложения с мултимедийни компоненти - Представа аргументирано разработено софтуерно приложение пред публика
------------	--

УЧЕБНО СЪДЪРЖАНИЕ:

Теми	Компетентности като очаквани резултати от обучението	Нови понятия / знания
1. ОСНОВИ НА ИНФОРМАТИКАТА		
1.1. Числата и техните представяния.	• Разширява и обобщава знанията, свързани с числата и техните представяния: непозиционни и позиционни бройни системи; формат на числата в десетична, двоична, четиритична, осмична и шестнадесетична бройна система. • Превръща числа от десетична в двоична бройна система и обратно. • Извършва събиране, изваждане и умножение на две числа в двоична бройна система. • Дава примери за използване на двоична и шестнадесетична бройна система.	• непозиционни бройни системи • позиционни бройни системи • десетична система • двоична система • четиритична система • осмична система • шестнадесетична система
1.2. Информационни дейности и процеси	• Обяснява предмета на информатиката и ролята ѝ в съвременното общество. • Изброява и описва основните информационни дейности събиране, съхраняване, преработка и разпространение и общата схема на информационните потоци. • Описва понятието информационен процес и дава примери на информационни процеси, свързани с решаване на житейски задачи. • Различава понятията информация и данни. • Обяснява и илюстрира с примери връзката между информация и данни. • Обяснява и илюстрира с примери същността на	• основни информационни дейности • информация, данни, дискретно представяне на информацията

	дискретното представяне на информацията за трансформирането ѝ в данни.	
1.3. Алгоритми и езици за програмиране	• Дефинира понятието алгоритъм и описва основните му характеристики (резултатност, крайност, детерминираност, масовост). • Описва и проследява несложни, линейни и разклонени алгоритми с различни средства. • Описва същността, структурата и разновидностите на цикличните алгоритмични конструкции. • Обяснява същността и функционалното предназначение на език за програмиране. • Обяснява същността и предназначението на транслятор (интерпретатор, компилатор). • Представа исторически факти, свързани със създаването и развитието на съвременните езици и среди за програмиране.	• алгоритъм • език за програмиране • транслятор • интерпретатор • компилатор
2. СРЕДА ЗА ВИЗУАЛНО ПРОГРАМИРАНЕ		
2.1. Интегрирана среда за визуално програмиране	• Отваря проект в интегрирана среда за визуално програмиране. • Редактира дизайн на графичния потребителски интерфейс на приложение в интегрирана среда за програмиране. • Запазва проект на приложение чрез средствата на интегрирана среда за програмиране. • Стартира приложение с графичен потребителски интерфейс чрез средствата на интегрирана среда за програмиране. • Разпознава основни компоненти на интегрирана среда за програмиране – графичен и текстов редактор, панел с контроли, панел за свойства на обект, панел за съобщения, панел за преглед на структурата на приложението.	• интегрирана среда за програмиране • свързващ редактор (linker) • програма за откриване и отстраняване на грешки (debugger) • редактор за проектиране на дизайн на графичен потребителски интерфейс

2.2. Основни етапи на създаване и изпълнение на компютърна програма	• Анализира задача с несложен математически модел. • Създава математически модел за решаване на несложна задача. • Съпоставя математически модел с програмно решение на даден проблем. • Открива основните компоненти на математически модел в демонстрирано програмно решение на даден проблем. • Стартира чрез средствата на интегрирана среда предварително подготвена компютърна програма с графичен потребителски интерфейс. • Тества предварително подготвен несложен проект. • Разпознава видовете грешки при програмиране. • Разчита и прави предположение за естеството на синтактична грешка в даден проект. • Открива и прави предположение за причината за логическа грешка в дадено приложение. • Открива и прави предположение за причината за грешка по време на изпълнение на приложение.	• синтактични грешки в компютърна програма • логически грешки в компютърна програма • грешки по време на изпълнение на програмата
2.3. Проектиране на графичен потребителски интерфейс	• Знае предназначението на основни контейнери и контроли – форма, етикет, текстово поле, бутон, диалогова кутия. • Разпознава основни свойства на графични обекти контроли – име, състояние, етикет, фон, настройка на шрифт и др. • Проектира несложна форма, съдържаща етикет, текстово поле, бутон. • Настройва основни свойства на форма, етикет, текстово поле и бутон. • Именува обекти-контроли съгласно общоприетата конвенция. • Задава функционалност на бутон, свързана с извеждането на статично съобщение в диалогова кутия.	• Интерфейсен компонент (контрола) • контейнер на контроли • свойство на обект • метод на обект

3. ПРОГРАМИРАНЕ		
3.1. Основни типове данни		
3.1.1. Тип низ	- Познава правила за именуване на константи и променливи. - Декларира, описва и инициализира променливи и константи от тип низ. - Присвоява стойност на променлива от тип низ. - Въвежда и извежда данни от тип низ в/от текстово поле. - Извежда данни от тип низ в/от етикет. - Извършва конкатенация на низове. - Използва стандартни методи на интерфейсни компоненти за форматиране на текст.	- символ - низ - множество на допустимите данни - множество на допустимите операции - име, тип стойност на променлива - име, тип и стойност на константа - присвояване на стойност - конкатенация
3.1.2. Целочислени типове данни	- Декларира, описва и инициализира променливи и константи от целочислен тип данни. - Използва вградени функции за преобразуване на низ в цяло число и обратното. - Въвежда и извежда данни от целочислен тип. - Използва различни целочислени типове данни. - Познава целочислените аритметични операции и техния приоритет. - Конструира аритметични изрази, съдържачи само целочислени данни, спазвайки синтаксиса и семантиката на конкретния език за програмиране. - Прилага и анализира резултатите от операциите събиране, изваждане, умножение, деление, цяла част и остатък от целочислено деление. - Реализира модел за решаване на задачи, базиран на целочислени типове данни.	- целочислен тип данни - конвенция за именуване на константи и променливи
3.1.3. Реални типове данни	Декларира, описва и инициализира променливи и	Реален тип данни

	константи от реален тип. • Използва вградени функции за преобразуване на низ в реално число и обратното. • Въвежда и извежда данни от реален тип данни. • Използва различни реални типове данни. • Познава приоритетите на аритметичните операции при реални типове данни. • Конструира аритметични изрази, съдържащи реални типове данни, спазвайки синтаксиса и семантиката на конкретния език за програмиране. • Прилага и анализира резултатите от операциите – събиране, изваждане, умножение, деление. • Реализира модел за решаване на задачи, базиран на реални типове данни.	
3.1.4. Аритметични изрази и вградени математически функции. Приоритет на операциите	• Оценява числената стойност на аритметичен израз, записан на език за програмиране. • Записва аритметичен израз със средствата на език за програмиране. • Прилага и използва вградени в езика за програмиране математически функции – абсолютна стойност, повдигане на степен, закръгляване, извличане на цялата част на реално число. • Използва приоритет на операциите в аритметични изрази, съдържащи вградени функции. • Създава аритметични изрази, съдържащи различни типове данни, като се съобразява със съвместимостта им. • Описва синтаксис и семантика на оператор за присвояване. • Форматира изхода на реално число.	• аритметичен израз в език за програмиране • вградени математически функции в език за програмиране • съвместимост на типове данни • форматиран изход
3.2 Създаване на компютърна програма за решаване на конкретна	• Знае основните етапи при създаване на компютърна програма. • Анализира и проектира решението на конкретна задача.	• коментари • оформяне на програмния код

задача	• Създава математическия модел за решаване на задачата. • Разработва алгоритъм за решаване на задачата. • Определя входно-изходни данни и техните типове. • Структурира и разработва графичен интерфейс, като използва обекти и декларира променливи. • Създава и описва програмния код. • Стартира, тества и валидира готовия проект. • Открива синтактични и логически грешки в програмата. • Отстранява синтактични и логически грешки при програмиране. • Спазва изисквания за оформление на програмния код, включващи подравняване, коментари, именуване на програмните единици.	
3.3. Програмни конструкции за реализация на разклонен алгоритъм		
3.3.1. Булев тип данни	Обосновава необходимостта от разклоняване на алгоритмичния процес. • Използва константите от булев тип данни. • Дава примери, в които се използва булев тип данни. • Декларира булева променлива. • Присвоява стойност на булева променлива. • Записва на език за програмиране булев израз, съдържащ операции за сравнение. • Изписва синтактично правилно на езика за програмиране основните логически операции – логическо отрицание, дизюнкция, конюнкция. • Познава приоритета на логическите операции.	• булев тип данни • булеви константи – false, true • логически операции • приоритет на логическите операции • булев израз

	• Пресмята без използване на компютър стойността на булев израз. • Съставя със средствата на език за програмиране сложен булев израз, отговарящ на дадена логическа ситуация.	• условен оператор • съставен оператор
3.3.2. Условен оператор	• Описва синтаксиса и семантиката на кратка и пълна форма на условен оператор. • Описва разклонен алгоритъм с помощта на условен оператор. • Използва условен оператор за проверка на коректността на входните данни за програма. • Използва условен оператор за обработка на свойства на радио-бутон и поле за отметка.	
3.3.3. Вложени условни оператори	• Обяснява семантиката на вложени условни оператори в кратка и пълна форма. • Проиграва изпълнението на фрагмент на програма, съдържаща вложени условни оператори. • Записва синтактично и логически правилно вложени условни оператори. • Създава модел и алгоритъм за решаване на задача чрез използване на вложени условни конструкции. • Реализира модел за решаване на задача чрез използване на вложени условни оператори. • Замена вложен условен оператор с единичен и обратно.	• вложен условен оператор
3.4. Програмни конструкции за реализация на циклични алгоритми		
3.4.1. Циклични алгоритми	• Посочва елементите на циклична конструкция – инициализация, тяло на цикъла и условие на цикъла. • Оценява необходимостта от използването на алгоритми с циклични конструкции с предусловие и постусловие. • Записва синтактично и логически правилно оператори за цикъл с предусловие, постусловие и управлявани от брояч.	• структура на циклични алгоритмични конструкции • оператор за цикъл с предусловие • оператор за цикъл с

	• Прилага алгоритми с циклични конструкции за проверка на входни данни. • Оценява необходимостта от използване на алгоритми с циклични конструкции с условие или управлявани от брояч. • Преобразува програмен код, съдържащ циклична конструкция, управлявана от брояч в циклична конструкция или управлявана от условие. • Открива синтактични и логически грешки в програмния код на алгоритми с циклична конструкция. • Прилага циклични алгоритми за управление на графичен потребителски интерфейс.	постусловие • оператор за цикъл, управляван от брояч • списъчно поле
3.4.2. Приложение на условни и циклични конструкции	• Прилага циклични алгоритми за изчертаване на графични примитиви. • Използва циклични алгоритми за въвеждане и извеждане на данни. • Прилага програмни конструкции за реализация на алгоритми за намиране на сума, минимум/максимум елемент, средно аритметично и др. в редици от числа, въвеждани от потребителския интерфейс/клавиатурата.	• графичен примитив • кутия за изображения
3.5. Тестване и верификация на програма	• Обяснява и разграничава понятията тестване и верификация. • Дефинира тестови данни • Дефинира очаквани резултати от тестването при определени входни данни. • Използва инструмент за откриване и отстраняване на грешки (debugger). • Прилага процедури за тестване и верификация на вече създадени програми.	• тестване • верификация • тестови данни
4. СЪЗДАВАНЕ НА СОФТУЕРЕН ПРОЕКТ		
4.1 Анализ на проекта	• Описва етапите при реализиране на софтуерен проект. • Избира клиент за своя проект и обосновава избора си / нуждата на клиента от такава разработка.	

	• Идентифицира и анализира нуждите на клиента • Разглежда и анализира други продукти, подобни на изборани, за да открие спецификите на това, което ще създаде.	
4.2 Изграждане на дизайн	• Поставя си конкретни цели, които да бъдат постигнати с разработването на проекта. • Създава модел за решаване на проблема, поставен в заданието на проекта. • Проектира графичен потребителски интерфейс.	
4.3 Създаване на продукт (писане на програмен код)	• Създава план с времево разпределение за създаването на разработката. • Демонстрира различни технически умения при разработване на проекта. • Следва създадения план в точка и създава програмен код за реализация на модела.	
4.4 Оценка на проекта и възможности за развитие	• Използва различни тестови техники, за да прецени дали създадения продукт работи коректно и отговаря на поставените цели. • Дава идеи за усъвършенстване на разработката си. • Изготвя документация за софтуерния проект. • Презентира и защитава готовия софтуерен проект.	

ПРЕПОРЪЧИТЕЛНО ПРОЦЕНТНО РАЗПРЕДЕЛЕНИЕ НА ЗАДЪЛЖИТЕЛНИТЕ УЧЕБНИ ЧАСОВЕ ЗА ГОДИНАТА:

Изисквания за провеждане на обучението:

1. Програмата предвижда обучението да се извършва на базата на език за визуално програмиране – Java. Разработена е така, че усвояването на необходимите понятия и умения да става чрез активно участие на ученика в учебния процес.

2. Препоръчва се обучението по първите три теми да се провежда предимно под формата на комбинирани уроци, с продължителност 2 последователни учебни часа. Уроците от този тип е подходящо да са със следната най-обща структура:

2.1. Упражнения за затвърждаване на учебното съдържание, съчетани с проверка и оценка на знанията и уменията и ориентирани (при възможност) към мотивация на новото учебно съдържание.

2.2. Мотивиране и въвеждане на новото учебно съдържание (теоретична и манипулативна компонента). Препоръчва се новото учебно съдържание да се дозираща и въвежда от учителя като последователност от смислово обособени и логически подредени части (неголеми модули). След всеки такъв модул се отдели учебно време за самостоятелна практическа работа на учениците, подпомогната от учителя.

2.3. Самостоятелна работа на учениците, свързана с усвояването на теоретичната и манипулативната компонента на учебното съдържание.

2.4. Анализ на хода и резултатите от урока и задаване на домашна работа.

3. Учебното време на комбинираните уроци по Тема 2 и Тема 3 съчетава въвеждането на нови знания с упражнения на компютър. Чисто теоретичните занятия в лекционен стил трябва да бъдат сведени до минимум. Обемът на практическата работа има превес над теоретичните части.

4. Обучението по Тема 4 предвижда учениците да разработват софтуерен проект, съдържащ проверка за коректност на входни данни, разклонени и циклични алгоритми, графичен интерфейс, форматиране на изход.

Препоръчително разпределение на часовете:

За нови знания и упражнения	30%
За обобщение, контрол и оценка	20%
Практически дейности	50%

СПЕЦИФИЧНИ МЕТОДИ И ФОРМИ ЗА ОЦЕНЯВАНЕ НА ПОСТИЖЕНИЯТА НА УЧЕНИЦИТЕ

Форми на оценяване:

- Устна

Устната форма на оценка е удачна при групови обсъждания, дискусии, генериране на идеи. Оценяват се идеите и аргументите на ученика. Не се осъществява устно изпитване, отнасящо се до запаметяване на теория.

- Писмена

Писмената форма е подходяща при проверка и оценка на теоретични знания и разбиране на синтаксиса и семантиката на даден оператор. Включват се въпроси с отворен отговор, задачи за изчисление на израз, задачи за проследяване на работата на алгоритъм или програмен фрагмент, задачи за откриване на логически грешки в алгоритъм или програмен фрагмент. Писмената форма на изпитване може да се провежда самостоятелно или в съчетание с практическа форма.

- Практическа

Тъй като предметът е с предимно практическа насоченост, това е преобладаващата форма в оценяването на постиженията на учениците. Практическо изпитване се прави на базата на портфолио, съдържащо резултатите от практическа работа в клас и разработка на проект в качеството му на изходно ниво от обучението. Оценка отразява всички етапи от практическата разработка (анализ, дизайн, програмиране, оценка и защита и др. според заданието), като за тази цел учителът подготвя съответните критериални матрици.

СЪОТНОШЕНИЕ ПРИ ФОРМИРАНЕ НА СРОЧНА И ГОДИШНА ОЦЕНКА:

Текущи оценки от работа в клас, участие в групови обсъждания и дискусии, писмено изпитване	35%
Текущи оценки от практически задания в клас	25%
Оценка на изходно ниво (проект).	40%

ДЕЙНОСТИ ЗА ПРИДОБИВАНЕ НА КЮЧОВИ КОМПЕТЕНТНОСТИ, КАКТО И МЕЖДУПРЕДМЕТНИ ВРЪЗКИ

Дейност 1.	Проучване и представяне на допълнителна информация, свързана с изучаваните теми. Учениците развиват умения за учене, социални и граждански компетентности, създават се междупредметни връзки с български език и литература.
Дейност 2.	Структуриране на статистическа информация във вид на данни в контекста на конкретна информационна задача. На учениците се дава възможност да демонстрират социални и граждански компетентности, както и компетентности в областта на българския език.
Дейност 3.	Идентифициране и дефиниране на стандартите в информатиката и използване на конвенции за именуване. Учениците демонстрират дигитални компетентности и умения за общуване на чужди езици.
Дейност 4.	Редактиране, компилиране и изпълнение на готова програма, създадена за различни сфери на обществения живот. Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности. Осъществяват се междупредметни връзки с природни науки, информационни технологии и математика.
Дейност 5.	Редактиране на дизайн на приложение с графичен потребителски интерфейс във визуална среда за програмиране. Реализират се междупредметни връзки на информационните технологии и изобразителното изкуство - цвят, цветови схеми, елементи на интерфейса и др.
Дейност 6.	Създаване на математически модели на ситуации, изучавани в часовете по математика и природни науки, като се използват средствата на програмирането. Учениците демонстрират математически и дигитални компетентности в областта на природните науки и на технологиите.
Дейност 7.	Настройки на основните свойства на графичните компоненти на приложение. Създаване на програма за изчисляване на печалбата на дадено предприятие, общия приход (разход) на домакинство. Демонстрира се компетентността инициативност и предприемчивост.
Дейност 8.	Описване на алгоритми за решаване на линейни уравнения, линейни неравенства и сравняване на отсечки и ъгли.

	Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки.
Дейност 9.	Въвеждане и извеждане на данни в приложения, свързани в природо-математическите дисциплини, бита и бизнеса. Класифициране на типовете данни. Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности. Осъществяват се междупредметни връзки с природните науки, информационните технологии и математиката.
Дейност 10.	Въвеждане и извеждане на данни от тип низ в текстово поле. Прилагат граматически правила в българския език за конструиране на изречение, работят с измерителни единици. Осъществяват се междупредметни връзки с български език и литература, математика, природни науки.
Дейност 11.	Създаване на аритметични изрази, съдържащи различни типове данни, отчитайки съвместимостта им. Осъществяване междупредметни връзки с математиката (пресмятане на числена стойност на израз, разлагане на множители), физиката (определяне на пътя и скоростта при равноускорително движение и свободно падане).
Дейност 12.	Реализиране на несложни алгоритми със средствата на език за програмиране. Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности.
Дейност 13.	Реализиране на програми с графично изобразяване на геометрични обекти и комбинация от обекти. Осъществяват междупредметна връзка с математика – създаване на програма за визуализация на задачи, свързани с пресмятане на точки на пресичане на геометрични обекти (отсечки, окръжности) и изобразяване на траектория, описана от точка, средна отсечка, триъгълник, симетрала.
Дейност 14.	Пресмятане стойността на булеви изрази чрез език за програмиране, обсъждане на съждения и техните отрицания. Осъществяват междупредметна връзка с математика. Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности.
Дейност 15.	Реализиране на алгоритми с използване на повторение, управлявано от условие. Осъществя се междупредметна връзка с математика – създаване на алгоритъм за намиране на най-голямо (най-малко) от няколко числа. Реализиране на програми за определяне на вида на триъгълник, определяне на дата по зададено условие, определяне на процент или други задачи от изучавани теми.

	Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности
Дейност 16.	Събиране на информация от различни източници. Трансформиране на информацията във формат на данни. Учениците развиват дигитални, математически компетентности и основни компетентности в областта на природните науки и на технологиите, социални и граждански компетентности.
Дейност 17.	Създаване на самостоятелно софтуерно приложение. Развиват се граждански компетентности - форми на поведение за успешно участие в социалния живот, разрешаване на конфликти и проблеми, умения за самостоятелно учене и събиране на информация. Разработват се общественополезни и екологични теми.