Физика и информатика

Увод

Развој информационих технологија уско је везан за развој саме физике. Једни од навећих напредака у информационој технологији настали су у тренуцима великих физичких открића, пред крај другог светског рата, 60-тих и 90-тих година прошлог века.

Иако ученицима изгледа да веза физике и информатике не постоји она је јако присутна. Област физике која се назива електродинамика (у основној школи то су области електрицитет и магнетизам) је основа електротехнике. Развојом информационих технологија решени су неки физички проблеми који су омогућили велика откиђа у физици која су потом применута у информационим технологија. Развој информатике и физике је уско повезан, пример је CERN (Европски центар за нуклеарна истраживања који се налази у Женеви). Пример ове сарадње је стварање нове информатичке области науке о великим подацима (Big data science).

Због растућег тренда уписа ученика на информатичке смерове како у средњим школама, тако и на факултету на овом месту је покушано да се ученицима омогући да решавају физичке проблеме помоћу информатике већ у основној школи.

По плану и програму основне школе у 6 разреду основне школе учи се програмски језик Python због чега је један примерак кода написан у овом програмском језику. Други примерак кода написан је у програмском језику C/C+++ који представља основни програмски језик у већини информатичких средњих школа као и програм у коме се програмира у вишим разредима основне школе на додатној настави.

Овај програм секције изводи се у сарадњи са наставницом информатике, на тај начин да теоријски део ученици раде са наставником физике, а практичан део реализацију у програмском језику са наставницом информатике. На овај начин је извршена корелација између информатике и физике.

Након завршетка програма секције ученици су у прилици да самостално решавају практичне физичке проблеме, задатке, да праве образовне програме који решавају физичке проблеме све уз коришћење информационих технологија.

Сви програми који настану из ове секције биће конзуларног типа, и моћиће се да скину са званичне странице школе.

Проблемски задаци

Да би се приступило практичном решавању проблема потребно је прво да се добро дефинише проблем који се жели решити.

Практични проблеми који ће се решавати из 6 разреда су:

- Одређивање једне од величина код равномерног праволинијског кретања ако су остале две познате.
- Одређивање једне од величине по формули за тежину тела код равномерно праволинијског кретања.
- Претварање јединица једних у друге (опште)
- Одређивање једне од величина по формули за густину
- Одређивање једне величине по формули за притисак

Практични проблеми који ће се решавати из 7 разреда су:

- Одређивање једне од величина код равномерно убрзаног праволинијског кретања ако су остале познате.
- Одређивање једне величине по формули за Други Њутнов закон, ако су остале две познате.
- Одређивање једне величине, ако су остале три познате по формули за двокраку полугу
- Одређивање да ли тело тоне, плута или плива у некој средини укуцавањем вредности његове густине.
- Одређивање једне величине, ако су остале две познате по формули за рад
- Одређивање једне величине, ако су остале две познате по формули за кинетичку енергију
- Одређивање једне величине, ако су остале три познате по формули за потенцијалну енергију
- Одређивање једне величине, ако су остале две познате по формули за снагу
- Одређивање једне величине, ако су остале две познате по формули за коефицијент корисног дејства
- Одређивање једне величине, ако су остале две познате по формули за количину топлоте
- Направити програм који врши конверзију температурних скала из целзијума у келвине и из келвина у целзијусе.

Практични проблеми који ће се решавати из 8 разреда су:

- Одређивање једне величине, ако су остале две познате по формули за период математичког клатна
- Одређивање једне од величине, ако су остале две познате по формули која даје везу између таласне дужине и фреквенције
- Одређивање једне величине, ако су остале познате по формули за индекс преламања
- Одређивање једне величине , ако су остале познате по формули за жижну даљину код сферног огледала или сабирног сочива
- Одређивање једне величине, ако су остале познате по формули за Кулонову силу
- Одређивање једне величине, ако су остале познате по формули за електрично поље
- Одређивање једне величине, ако су остале познате по формули за потенцијалну
- Одређивање једне величине, ако су остале познате по формули за напон
- Одређивање једне величине, ако су остале познате по формули за јачину струје

- Одређивање једне величине, ако су остале познате по формули за Омов закон за део кола
- Одређивање једне величине, ако су остале познате по формули за отпорност проводника
- Одређивање једне величине, ако су остале познате по формули за Омов закон за цело коло
- Одређивање једне величине, ако су остале познате по формули за Лоренцову силу
- Одређивање једне величине, ако су остале познате по формули за Амперову силу
- Одређивање једне величине, ако су остале познате по формули за магнетну индукцију
- Одређивање једне величине, ако су остале познате по формули за активност код радиоактивног зрачења

Принцип самосталног решавања задатка

Након дефинисања проблема од интереса потребно је даље потражити теоријска објашења везана за дати проблеме. То се ради тако што отворимо књигу из физике и потражимо лекцију у којој се обрађује дата проблематика, прочитамо је колико год пута је потребно док год у потпуности до најситнијих детаља не разумememo проблемску ситуацију. Након тога опточињемо процес планирања како би најбоље било решити задатак и уз то постићи што већу општост. Након чега се иде у реализацију проблема у неком од програмских језика (Python или C++).

Алгоритам

Алгоритам представља упутство које програмер треба да прати да би правилно направио програм који би решавао задати проблем. Посао прављења алгоритма сам по себи представља уметност, пошто је потребно око 10 година искуства у практичним пројектима у једној области да би програмер био спреман да прави самостално алгоритме. Разлика између добрих и лоших програм настаје услед лошег алгоритма.

Најбоља аналогија са алгоритмом представља рецепт за кување. Ако се кувар придржава рецепта он ће увек направити јело тог укуса, а ако неискусни кувар почне да експериментише велике шансе су да направи безукусно јело које нико неће желети да једе.

Кораци у формирању алгоритма

Први корак: Сагледати проблем од горе ка доле

Под овим се подразумева да програмер разуме до најситнијих детаља проблематику

Други корак: Направити стратегију

Ово је један од најкритичнијих делова проблема прављења алгоритма, пошто је потребно направити анализу техничких захтева, ради добијања оптималног решења. У овој фази се бира програмски језик у ком се жели испрограмирати, анализирају се предности и мане одређених функција, лакоћа реализације, радни сати и на крају брзина извршавања програма.

Трећи корак: Доношење одлуке о писању алгоритма

Отпочиње процес писања упутства на који начин треба вршити програмирање и у ком

програму.

Четврти корак: Последња провера

Овде се узима коначан алогритам и још једном сваки корак његов се проверава ради проналажења грешака или његовог евентуалног унапређења.

Пети корак: Предаја алгоритма

Како модерне програме не прави само један програмер, човек који је направио алгоритам сад са њим упознаје друге програмер и предаје им задатке шта ко треба да уради.

Пример: Одређивање једне од величина код равномерног праволинијског кретања ако су остале две познате

Теоријски део

Равномерно праволинијско кретање представља кретање у коме се тело креће по правој линији константном брзином. То значи да се може применити на следеће проблемске ситуације: кретање аутомобила по правом путу при чему се креће максималном дозвољеном брзином коју не мења током одређеног дела пута, кретање авиона, кретање брода, трчање на 100 метара...

Формула по којој се решава задатак је:

$$v = \frac{s}{t}$$

при чему је

v – брзина тела
s – пређени пут
t – протекло време

- Брзина тела мора бити дата у m/s ако желимо да убацујемо вредност у формулу, ако то није случај врши се претварање.
- Пређени пут мора бити дат у метрима, пре убацивања у формулу, ако то није случај вршимо претварање
- Протекло време мора бити дато у секундама, пре убацивања у формулу, ако то није случај вршимо претварање

У зависности да ли нам је потребно да вредност непознате величине буде дата у основној или некој другој јединици на крају постоји могућност да нам је потребно да извршимо претварање.

Код програм у C/C++ и Python

Код програма се налази у фајлу RPK.cpp, а унапређена верзија RPK1.cpp које је могуће добити, ако се ученици обрате наставници информатике или могу скинут са сајта школе. Корекције које су урађене у унапређеној верзији су у томе да се користе компликованији алати за програмирање, тако да је потребно да ученик више изучава програмски језик да би по овом систему направио код. Потребно је да познаје процедуралну и функционалну парадигму програмирања у програмском језику.

Код у програмском језику Python дат је у фајловима RPK.py, а унапређена верзија у RPK1.py фајлу. Поново за разумевање унапређене везије потребно је познавати начин на који се реализује процедурална и функционална парадигма у програмском језику.

У RPK.cpp или RPK.py фајлу приступило је се решавању проблема на један веома баналан начин. Овде је знање програмског језика сведено на основне улазно излазне функције као и основне математичке функције које су све уграђене у програмском језику. Иако једноставно направљен програм он ипак врши своју функцију. Овакава начин програмирања где се не гледа да се постигне што већа општост у решавању него да се само реши проблем специфичан је за инжењере и научнике којима је битно само да реше проблем. Док софтверски инжењери овакав тип програмирања сматрају бескорисним пошто делове овога програм не могу да користе у даљем програмирању других програма.

Унапређена верзија програма је захтевнија са становишта знања које је потребно имати везана за програмски језик. Овај прилаз софтверски инжењери јако воле пошто појединачне делове програм могу користи за прављење других програма, а самим тим уштедити себи време при програмирању.

На крају који прилаз је најбоље искористити зависи пре од доста фактора, због чега су оба прилаза приказана, а на ученицима је да самим изаберу пут којим желе да крену.

Савет за ученике је да изаберу једна програмски језик и да у њему пишу програме све док све основне не науче. Није препоручљиво прелазити на нови програмски језик док се опште градиво не научи, потом је прелаз са једног на други програмски језик поједностављено.