Сумський державний педагогічний університет імені А. С. Макаренка
Фізико-математичний факультет
Кафедра інформатики

УДК 378.016:51:004

Носаченко Дар'я Сергіївна

ОСОБЛИВОСТІ РОЗВ'ЯЗУВАННЯ КОМБІНАТОРНИХ ЗАДАЧ НА ОЛІМПІАДАХ З ПРОГРАМУВАННЯ

Галузь знань: 01 Освіта
Спеціальність 014 Середня освіта (Інформатика)
Кваліфікаційна робота на здобуття освітнього рівня «Магістр»

Науковий керівник:
_____ В.Г. Шамоня,
кандидат фізико-математичних наук,
доцент кафедри інформатики

Виконавець:
_____ Д.С. Носаченко

Суми 2021

ЗМІСТ

ВСТУП

Актуальність дослідження. Україна займає одне з основних місць стосовно підготовки ІТ-спеціалістів. Саме тому, талановиті діти, що зможуть реалізувати себе в даній галузі є важливими для суспільства. Вчителю необхідно вміти знаходити таких дітей та допомагати їм у їх розвитку.

Вміння розв’язування задач є одним з головних показників рівня математичного розвитку учня, глибини опанування навчального матеріалу.

Аналізуючи концепцію "Державної програми роботи з обдарованою молоддю" завданням сучасної школи є не тільки значне засвоєння та відтворення отриманих знань, але і розвивання пошуково-пізнавальних здібностей школярів, формування всебічно розвинутої особистості з компетентностями, необхідними для подальшого життя, з метою їх успішної соціалізації у суспільстві та високої здатності до самонавчання. Також в даній концепції зазначено завдання залучення обдарованої молоді до олімпіад, конкурсів і т.д. [57]

Вчитель, знайшовши такого обдарованого школяра, в якого є потенціал та мотивація працювати над розв’язуванням олімпіадних задач з програмування, має правильно спланувати спільну роботу першочергово, і до того ж визначити шляхи навчання дитини новим знанням з розв’язування нестандартних задач, беручи до уваги тенденції сучасного навчання, організувати залучення учня до самоосвіти, при цьому будучи консультантом і наставником дитини.

Аналіз актуальних досліджень в галузі дидактики показав ряд робіт, які також аналізують проблему організації підготовки учнів до олімпіад з інформатики, включаючи також розгляд комбінаторних задач. Проблеми підготовки до олімпіади в свої працях розглядали науковці Жуковський С. С., Горошко Ю. В., Міца О. В., Атанов Г. О., Лапінський В., та ін. Проблемою сервісів для підготовки до олімпіади займалися Франчук Н. П., Ковальчук М. М., Колесник Н., та ін. Питання використання комбінаторних задач в підготовці до олімпіад з програмування

розглядали в своїх працях Жуковський С. С., Дідковський В. Л., Матвійчук С. В., та ін.

Разом з тим, через швидкий розвиток інформаційних технологій, мов програмування, появи нових мов та забуття старих, ускладнення та розширення тем для олімпіад з інформатики та шкільного програмування дозволяють говорити про актуальність проблеми підготовки до олімпіади з програмування.

Об'єкт дослідження: олімпіади з програмування.

Предмет дослідження: методичні особливості підготовки до олімпіад з програмування, розглядаючи розв'язування комбінаторних задач мовою програмування Python.

Мета дослідження: визначити та проаналізувати особливості в підготовці до олімпіад з програмування, розв'язуванні комбінаторних задач та розробити приклади комбінаторних задач для підготовки до олімпіад з інформатики та уроки з використанням задач олімпіадного характеру для учнів ЗЗСО.

Поставлена мета дослідження обумовила вирішення низки **завдань**:

1) ґрунтуючись на науково-педагогічних джерелах охарактеризувати підготовку до олімпіади з інформатики, визначити основні її проблеми;

2) проаналізувати та охарактеризувати основні елементи комбінаторики: перестановки, сполучення та розміщення, створити коди мовою програмування Python чи описати розв'язок для реалізації знаходження їх кількості чи представлення їх всіх можливих у коді;

3) уточнити особливості та визначити шляхи для оцінки ефективності та налаштування алгоритму;

4) здійснити аналіз завдань олімпіад з програмування минулих років на наявність завдань з використанням елементів комбінаторики;

5) проаналізувати сервіси для підготовки до олімпіади з інформатики;

6) розробити приклади комбінаторних задач з програмування та розв'язки до них мовою програмування Python.

7) зробити аналіз навчальних програм стосовно наявності тем програмування та алгоритми та розробити конспекти уроків з використанням комбінаторних задач для підготовки до олімпіади з програмування.

Для досягнення мети використано низку **методів** дослідження:

теоретичні – аналіз та узагальнення науково-методичної літератури для доведення актуальності підготовки до олімпіад та використання для цього комбінаторних задач; термінологічний аналіз для уточнення основних понять олімпіад з програмування та комбінаторики; структурно-логічний аналіз для уточнення методичних особливостей організації підготовки до олімпіад з інформатики; контент-аналіз з метою характеристики сервісів для підготовки до олімпіад з програмування, аналізу напрацювань вчителів інформатики щодо підготовки до олімпіад з інформатики;

емпіричні – спостереження за освітнім процесом в ЗЗСО.

Практична значущість дослідження полягає в готовності розроблених прикладів, кодів та конспектів уроків до застосування в освітньому процесі для вивчення теми програмування.

Апробація матеріалів дослідження здійснювалася на наукових заходах, а саме: Студенська звітна конференція 2021 Фізико-метематичний факультет СумДПУ ім. А.С. Макаренка. «Матеріали результатів досліджень молодих науковців. Випуск 15. Том 1» (2021 рік, м. Суми) [75] та Студенська звітна конференція 2021 Фізико-метематичний факультет СумДПУ ім. А.С. Макаренка. «Матеріали результатів досліджень молодих науковців. Випуск 15. Том 2» (2021 рік, м. Суми). [76]

Робота буде цікава для вчителів інформатики та майбутніх вителів інформатики, які вмотивовані працювати з обдарованими учнями та брати активну учать у підготовці їх до олімпіад.

Структура та обсяг роботи. В даній науковій роботі 2 розділи, 24 рисунки, 11 формул, загальний обсяг сторінок – 58, кількість використаних джерел – 28.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ЗАСАДИ ПІДГОТОВКИ ДО ОЛІМПІАДИ З ІНФОРМАТИКИ ТА ПОНЯТТЯ ТИПУ КОМБІНАТОРНИХ ЗАДАЧ НА ОЛІМПІАДІ З ІНФОРМАТИКИ

1.1. Положення про всеукраїнські олімпіади з програмування

Однією із головних ознак ступеня математичного розвитку людини, рівня засвоєння навчального матеріалу є вміння розв'язувати задачі. У шкільних курсах математики, фізики і інформатики вивченню розв'язування задач приділяється велика кількість часу, але головний метод даного навчання являє собою демонстрацію наявних способів розв'язування деяких типів задач, і майже не даються потрібні знання аналізу суті задачі і її розв'язку.

Таким чином в школярів не виробляються уміння і навички в діях, що входять в основну діяльність з розв'язування задач, не стимулюється невпинний аналіз учнями своєї діяльності у цьому напрямку, по виокремленню в ній основних методів і підходів, що дало б змогу, у майбутньому, будувати власний план дослідження і розв'язання задач певного класу. [11, с. 85]

Відповідно до «Положення про Всеукраїнські учнівські олімпіади, турніри, конкурси з навчальних предметів, конкурси-захисти науково-дослідницьких робіт, олімпіади зі спеціальних дисциплін та конкурси фахової майстерності» серед учнів закладів загальної середньої освіти всеукраїнські учнівські олімпіади з інформатики проводяться щороку.

Олімпіади проводяться в чотири етапи. Перший етап – шкільний (міжшкільний) на базі закладів загальної середньої освіти, другий етап – районний (міський), третій етап – обласний, четвертий етап – на державному рівні. Перші три етапи олімпіади з інформатики проводяться серед учнів 8-11 класів, четвертий – серед учнів 9-11 класів.

Практичні олімпіадні завдання з інформатики, створюються таким чином, щоб в учасників були рівні можливості їх виконання та були умови повного виявлення кожним учнем його рівня знань та вмінь.

Для практичних турів олімпіади передбачено використання лише обладнання, програмного забезпечення, друкованих матеріалів, наданих оргкомітетом та журі. [25]

Стосовно підготовки обдарованих школярів до участі в олімпіаді з інформатики учні мають вивчати не лише мову і методи програмування.

Також варто врахувати, що результат залежить ще й від підготовки учнів до стресової ситуації, їх вмінні концентруватися на завданні і його виконанні та розподілу часу на завдання, знаходження нестандартних розв'язків завдань. [11, с. 7]

У розвитку вміння розв'язувати задачі з програмування вчителі найчастіше дають базу прикладів розв'язання певних типів (груп) задач, хоча варто було б давати учням знання та вміння аналізу, загальні методи і підходи розв'язання задач і вміння активного використання цього.

В шкільному курсі інформатики в розв'язуванні задач олімпіадного характеру передбачено тільки демонстрацію готових розв'язків задач і надано небагато часу на це.

Вже давно є розроблені схеми розв'язання олімпіадних завдань з інформатики. Однак є чинники, які суттєво ускладнюють їх використання на олімпіадах: своєрідний запис умови, відволікаючий від головних компонентів, потреба створення нових математичних моделей чи нестандартне використання і поєднання звичних. [5]

Використання таких схем можливе в підготовці учнів до олімпіади, але з удосконаленням, використанням інших методів і засобів.

Завдання олімпіадних задач з інформатики є в тому, щоб написати розв'язки певних алгоритмічних завдань однією з дозволених мов програмування.

Розв'язками задач є практично реалізовані програми, які виконують поставлені завдання для заданих вхідних даних, що відповідають вказаному формату та виводять отримані результати. Вищевказане створює умови для автоматизування процесу перевірки учнівських робіт і збульшує об'єктивність результатів оцінювання. [11, с.18-20]

Основними проблемами, представленими для розв'язання в змісті олімпіадних завдань, є математичні чи логічні. Здебільшого завдання для вирішення відносяться до наступних категорій: теорія чисел, теорія графів, геометрія, комбінаторика, аналіз рядків та структури даних. [9, с.14]

Підготовка до олімпіади з інформатики обов'язково потребує поділу задач для підготовки на групи. Прикладом поділу може бути наступний:

- комбінаторика;
- сортування та пошук;
- алгоритми на графах;
- алгоритми з довгими числами (довга арифметика);
- перебір та методи його скорочення;
- елементарні обчислення;
- елементи обчислювальної геометрії;
- динамічне програмування;
- теорія гри. [11]

Або С. В. Матвійчук [8] визначає наступні розділи:

- комбінаторику;
- арифметику цілих чисел;
- пошук і сортування;
- математичне моделювання;
- довгу арифметику цілих та дійсних чисел;
- жадібний алгоритм;

- алгоритм на графах;
- перебір та методи його скорочення;
- рекурсивні алгоритми;
- динамічне програмування;
- метод гілок та меж;
- метод «хвилі»;
- аналітичну геометрію.

Надалі мова йтиме про задачі олімпіадного характеру, що в своєму розв'язку використовують елементи комбінаторики: кількість перестановок, сполучень чи розміщень, або безпосереднє їх представлення чи інші базові елементи комбінаторики.

1.2. Комбінаторні задачі на олімпіадах з інформатики

У шкільному курсі інформатики розв'язуванню задач олімпіадного характеру майже не приділяється уваги. Багато вчителів зазначають, що в розв'язуванні олімпіадних задач важливий досвід, чим більше задач ви знаєте, тим краще справитесь з олімпіадою.

Задачі з програмування потребують відразу точно усвідомити умову, створити модель розв'язку, даної задачі, дослідити аналітично отриману модель, побудувати алгоритм і створити програму, для розв'язку задачі для будь-яких вхідних даних. Можна виділити шість базових етапів роботи над олімпіадною задачею.

Першим етапом є аналіз безпосередньо умови задачі. Умови олімпіадних задач з програмування складаються з наступних елементів: зміст, завдання, технічні умови, приклад вхідних та вихідних даних. В змісті знаходиться опис ситуації, що досліджується в цій задачі.

Олімпіадні задачі зазвичай мають відносно громіздкий зміст, інколи це заплутує учасника, чи зміст, який лише поетизує умову задачі та такий, який містить необхідну для розв'язання інформацію. Необхідно ретельно прочитати умову, вчитуючись у кожне слово. Підказка до розв'язку задачі може також бути прихована у формуванні вхідних і/чи вихідних даних. Без даних частин умови задача часом може мати цілковито не той зміст. Через це, варто звернути увагу на формулювання та приклади введення і виведення даних, а також на їх обмеження. Важливо навчити школярів виділяти основні елементи в умовах задач, не пропускаючи жодних деталей. [20]

Мета другого етапу – скласти план розв'язку задачі. При підготовці до олімпіади з інформатики тут необхідно навчити школярів ставити запитання до задачі, які допоможуть учню визначити метод її розв'язання. Запитання орієнтовно мають бути наступними. Чи була розв'язана аналогічна задача? Чи є задача, до якої задача може бути зведена? Якщо відповідь позитивна, то, очевидно, потрібно звести задачу до задачі, що була розв'язана раніше. Чи всі дані задачі були використані? Знаходження неврахованих даних спрощує складання методу її розв'язування. Чи можна визначити певні окремі випадки? [20]

В третьому етапі необхідно побудувати математичну модель, схему розв'язку. Побудова математичної моделі означає математичний опис процесів, фактів умови задачі чи знайти розв'язок, що працюватиме при відсутності обмежень пам'яті, часу, діапазону змінних та при відсутніх втратах точності змінних. Даний розв'язок звичайно буде неефективним, проте неефективний розв'язок означає розуміння умови задачі. За побудовою математичної моделі йде складання алгоритму. [20]

Четвертим етапом є безпосередньо сама реалізація алгоритму. Олімпіада з інформатики передбачає деякі бали також за задачу лише частково розв'язану. Через це потрібно здавати всі можливі розв'язки, нехай і немає впевненості в тому, що код є ефективним та пройде тести. [20]

На п'ятому етапі проводиться тестування коду. Після компіляції програми чи запуску коду на виконання (дивлячись яка мова програмування використовувалася), потрібно перевірити правильність роботи коду. Першочергово перевіряється правильність роботи на даних, що вже були в умові задачі. За потреби виправляються помилки, далі варто зробити деяку кількість невеликих тестів, що передбачають різні аспекти задачі, та представлення яких відомо. [20]

Шостим етапом є здача розв'язку. Декілька останніх разів олімпіади з програмування проводяться дистанційно. При виконанні завдань з олімпіади школярі мають змогу здати на перевірку розв'язок та перевірити системою його на тесті з умови задачі. Така можливість зменшує кількість учнівських помилок, що пов'язані з неправильним форматом зчитування даних.

Безпосередньо перед здачею коду розв'язку задачі потрібно звірити правильність формату вхідних та вихідних даних, попередити можливий вихід за межі масиву. На даному етапі помилка може означати невірний результат при правильно складених інших частинах коду. Проте може бути і неправильно виведений окремий блок чи зайві дані. [20]

Враховуючи все вищевказане, при підготовці до олімпіади з інформатики потрібно учнів навчити не лише методам розв'язування задач і основним алгоритмам, але й вчити правильно визначати час, потрібний для розв'язання тієї чи іншої задачі, виділяти етапи в процесі розв'язку задачі та дати розуміння кожного етапу.

Комбінаторика – це наука, в якій вивчаються сполуки (сукупності елементів, що розташовані за певними правилами). [20, с. 42-65]

Комбінаторика вивчає кількість комбінацій підпорядкованих визначеним умовам, які можна складати з елементів заданої скінченної множини.

Базовими поняттями при вивченні комбінаторики в школі є перестановки, розміщення та сполучення. [24]

Перестановкою з n-елементів називають будь-яку впорядковану n-елементну множину.

Перестановками називають комбінації з одних і тих самих елементів, що відрізняються тільки порядком їх розміщення. Формула перестановки – (формула 1.1). [26]

$$P(n) = n!$$

Формула 1.1

Елементи в комбінаціях можуть повторюватись і відповідно є формула для усунення повторень (формула 1.2).

$$P(n) = \frac{n!}{k_1! \cdot k_2 \cdot \ldots \cdot k_m!}; k_1 + k_2 + \cdots + k_m = n$$

Формула 1.2

Упорядкована m-елементна підмножина n-елементної множини називається розміщенням з n елементів по m елементів і позначається A_n^m. Розміщеннями називають комбінації з n елементів по m елементів, які відрізняються або самими елементами, або їх порядком. Формула знаходження кількості розміщень (формула 1.3).

$$A_n^m = \left(\frac{n!}{(n-k)!}\right)$$

Формула 1.3

Якщо елементи повторюються, то на кожне з місць є n варіантів вибору отже (формула 1.4).

$$A_n^m = n^m$$

Формула 1.4

Сполученнями називають будь-яку m-елементну підмножина n-елементної множини C_n^m. Сполученнями називають комбінації із n елементів по m елементів, які відрізняються хоча б одним елементом (відрізняються складом, порядок не суттєвий). Формула знаходження кількості сполучень (формула 1.5).

$$C_n^m = \frac{n!}{m!\,(n-m)!}$$

Формула 1.5

У випадку повторень кількість сполучень з повтореннями з n елементів по m елементів можна розглядати як кількість перестановок з повтореннями m+(n-1) (формула 1.6) (сполучення з повтореннями K).

$$K_n^m = C_{n+m-1}^m \frac{(n+m-1)!}{m!\,(n-1)!}$$

Формула 1.6

В комбінаториці використовуються два основні правила: Правило суми та Правило добутку.

Правило суми: якщо деякий елемент A з певної сукупності елементів можна вибрати m способами, елемент B можна вибрати n способами, то елемент "A або B" можна вибрати n+m способами. [26]

Правило добутку: якщо деякий елемент А з певної сукупності елементів можна вибрати m способами і після кожного такого вибору елемент В можна вибрати n способами, то елемент "А і В" можна вибрати m*n способами. [26]

Складання алгоритмів для розв'язання практичних задач часто зводиться до використання саме базових понять комбінаторики: вибрати деяку кількість елементів з певної скінченної множини елементів, які мають задані властивості, розмістити елементи множини у певному порядку, визначити всі підмножини елементів тощо. Такі задачі і називаються комбінаторними.

Для підготовки до олімпіади з інформатики доцільно розглядати тільки окремі елементи комбінаторики, які найчастіше використовуються в розв'язанні алгоритмічних задач та відносяться до визначення і підрахунку кількості деяких елементів підмножин із заданої кінцевої множини елементів.

Також варто розглядати не лише задачі алгоритмізації задач, розв'язок яких обмежується використанням комбінаторики, а й задачі, що використовують елементи комбінаторики як фрагменти складнішого алгоритму.

Розглядаючи алгоритмізацію задач з використанням базових понять комбінаторики (перестановок, розміщень, сполучень) варто звертати увагу не лише на саму суть понять та звичне їх використання, а і генерування цих базових понять. Відповідно можна розділити задачі на типи.

Розглядаючи задачі на перестановки можна виділити два варіанти розвитку подій. Перший випадок - в задачі вимагається підрахунок кількості можливих перестановок деякої множини з n елементів, в такому випадку вирішення задачі просте: варто лише запрограмувати обчислення n!.

Складнощі в цих задачах можуть виникнути лише з занадто великими числами, що варто передбачити визначаючи тип змінних, що використовуються в алгоритмі. Якщо використовувати мову програмування Python потрібно лише підключити модуль «math» та використати функцію «math.factorial» до потрібної змінної чи числа.

Другий варіант розвитку подій вимагає детального розгляду. В даних задачах потрібно знайти не кількість перестановок, а безпосередньо їх самі, тобто саме представлення всіх можливих розміщень n елементів по n елементів.

Для вирішення даних задач вводиться поняття “лексикографічний порядок перестановок”, що на множині всіх перестановок визначається: кажуть, що P1<P2 тоді і тільки тоді, коли існує таке t≥1, що P1[t]<P2[t] та P1[i]=P2[i] для всіх i<t.

Визначаючи всі можливі перестановки власне з’ясовуються на кожному наступному кроці слідуючі за лексикографічним порядком перестановки.

Якщо розташувати елементи множини перед початком роботи за зростанням та використовуючи лексикографічний порядок в кінці отримати розташовані елементи за спаданням, отримуються всі варіанти перестановок елементів. Отже задача на знаходження всіх можливих перестановок є задачею на визначення наступних значень множини у лексикографічному порядку. [13 с. 21-33]

Після даних висновків можна переходити до визначення побудови алгоритму даного типу задач.

Але, так як не завжди комбінаторні задачі пов’язані з числами, доцільно перед розв’язанням задачі визначити деяку ознаку, за якою можна визначати відношення між елементами заданої в умові множини.

Алгоритм створення лексикографічної послідовності елементів заданої скінченної множини:

- Визначити початковий стан даної множини елементів. Розташувати елементи за зростанням, у випадку коли це не так.
- Здійснювати рух даною послідовністю елементів справа наліво, виконуючи аналіз поточних елементів. Перехід до наступного елемента робиться тільки у випадку, коли перехід виконується до елемента, який менший від попереднього. Таким чином визначається граничний елемент.

- Починати рухатись знову від останнього елемента справа наліво, здійснюючи порівняння кожного обраного елемента зі значенням граничного, який визначився у пункті 2. Припинення руху відбувається, коли з'являється елемент, що є більшим за значенням від граничного. [13 с. 21-33]
- Елементи з пунктів 2 та 3 поміняти місцями.
- Елементи, які розташовані за граничним, розміщуються у зворотному порядку.

Приклад коду мовою програмування Python для цілих чисел (Рис.1.1), приклад його виконання для масиву [1, 2, 3, 4] (Рис. 1.2). Для переробки даного коду для перестановки текстових даних потрібно замінити рядок «a=[int(i) for i in input().split()]» на a=[v for v in input().split()]. Приклад показано в середовищі IDLE.

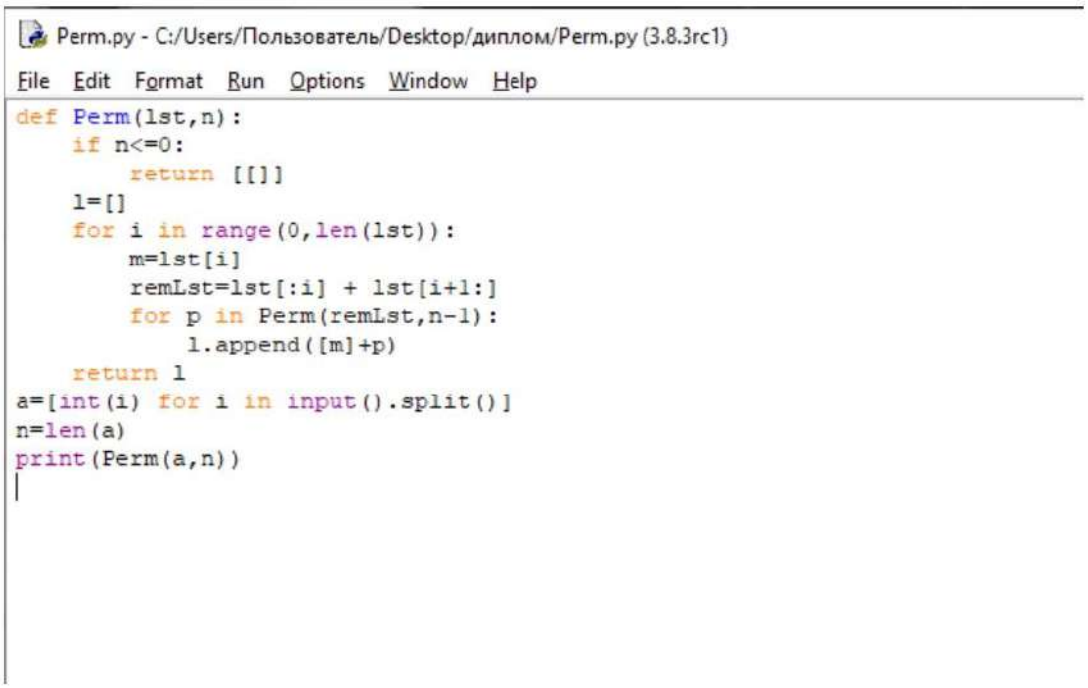

Рис. 1.1. Код для перестановок

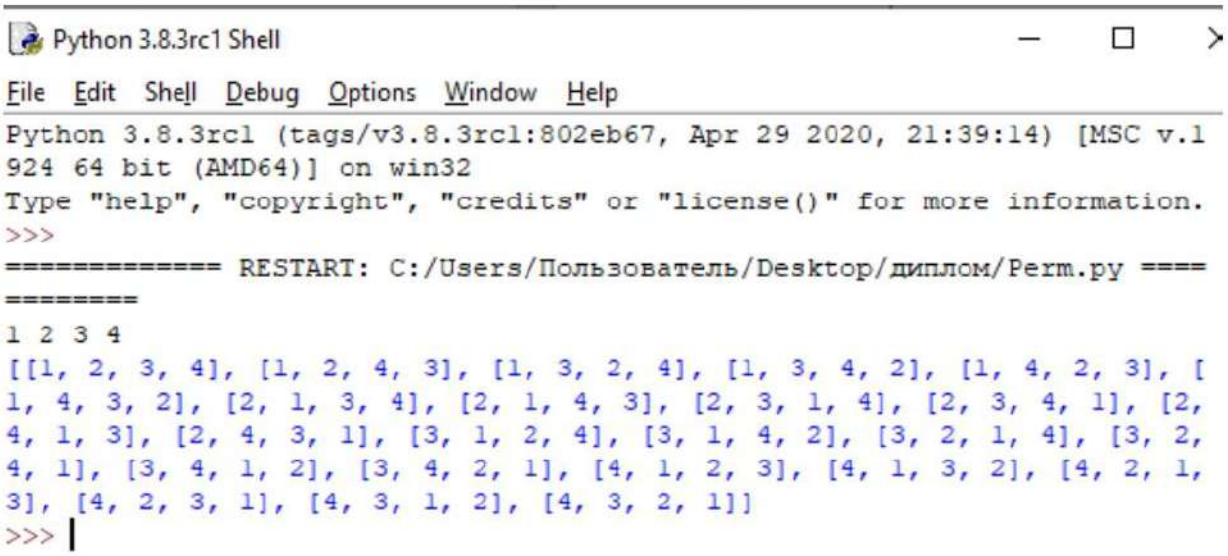

Рис. 1.2. Приклад виконання коду для перестановок

Наступними задачами, які варто розглянути – задачі з використанням сполучень.

Аналогічно перестановкам є варіант задач, в яких достатньо використати лише формулу для знаходження кількості сполучень, і відповідно.

Проблемою даних задач є лише значне зростання кількості сполучень при відносно незначному збільшенні значення n, проте для деяких мов програмування це не є проблемою.

Приклад коду для знаходження кількості сполучень мовою програмування Python з використання вбудованого модулю «math» (рис. 1.3) та приклад виконання даного коду розв'язку завдання знаходження кількості сполучень (рис. 1.4).

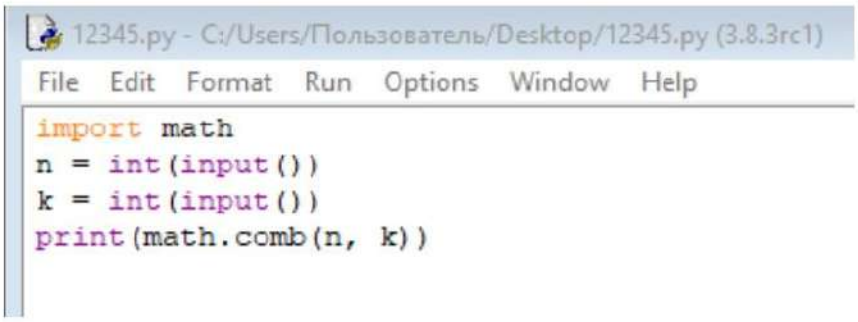

Рис. 1.3. Приклад коду для кількості сполучень

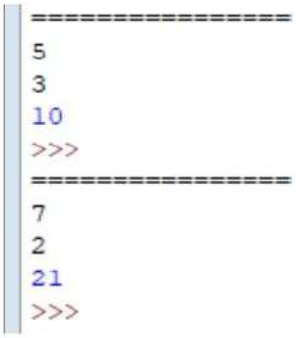

Рис. 1.4. Приклад виконання коду для кількості сполучень

У випадку, коли в умові задачі вимагається не лише визначення кількості сполучень, а й знаходження самих всіх можливих сполучень. У випадку сполучень попередній алгоритм зазнає значних змін.

Перед роботою алгоритму генеруємо всі сполучення з n (від 0 до n-1) цілих чисел по k.

Далі першим кроком алгоритму буде створення послідовності розміром k+2 елементи. Перші k-елементів визначаємо рівними їх індексу. Елемент k+1 позначаємо рівному n, k+2-ий елемент рівний нулю.

Наступним кроком, починаючи з початку послідовності перевіряємо наступну умову: $k_i+1 = k_{i+1}$. При рівності цих елементів встановлюємо елемент рівний його індексу. При порушенні умову відбувається перехід до наступного кроку.

Якщо індекс елементу, в якому порушилась попередня умова, більше k алгоритм завершується в іншому випадку збільшуємо елемент, з відповідним індексом, на одиницю. Повертаємося до попереднього кроку.

Приклад коду мовою програмування Python (рис. 1.5) та виконання цього коду (рис. 1.6).

Однак, даний код потребує доопрацювання: реалізація для будь-якої послідовністю з числами, різниця між якими рівна не лише одиниці.

```
def combination(k, n):
    c = []
    for i in range(k):
        c.append(i)
    c.append(n)
    c.append(0)
    while True:
        print (c[0:k])
        for j in range(len(c)-1):
            if c[j]+1 == c[j+1]:
                c[j] = j
            else:
                break
        if j<k:
            c[j] += 1
        else:
            break
k = int(input())
n = int(input())
print(combination(k, n))
```

Рис. 1.5. Код для сполучень

```
>>>
========= RESTART: C:
3
5
[0, 1, 2]
[0, 1, 3]
[0, 2, 3]
[1, 2, 3]
[0, 1, 4]
[0, 2, 4]
[1, 2, 4]
[0, 3, 4]
[1, 3, 4]
[2, 3, 4]
```

Рис. 1.6. Приклад виконання коду для сполучень

Далі варто визначити можливості одержання значення кількості розміщень, та знаходження безпосередньо самих розміщень.

Відповідно до двох попередніх типів задач, поділ задач та алгоритми їх розв'язання схожі.

Кількість розміщень – визначається відповідною формулою комбінаторики, з використанням модулю math, для обчислення факторіалу.

Код буде відрізнятись від перестановок лише тим, що n буде визначатись не довжиною.

Приклад коду для знаходження всіх можливих розміщень мовою програмування Python (рис. 1.7) та виконання даного коду на представлення усіх можливих розміщень коду для масиву з чотирьох елементів – [1, 2, 3, 4] розміщення з чотирьох по три елементи (рис. 1.8).

```
def Prod(lst,n):
    if n<=0:
        return [[]]
    l=[]
    for i in range(0,len(lst)):
        m=lst[i]
        remLst=lst[:i] + lst[i+1:]
        for p in Prod(remLst,n-1):
            l.append([m]+p)
    return l
a=[int(i) for i in input().split()]
n=int(input())
print(Prod(a,n))
```

Рис. 1.7. Код для розміщень

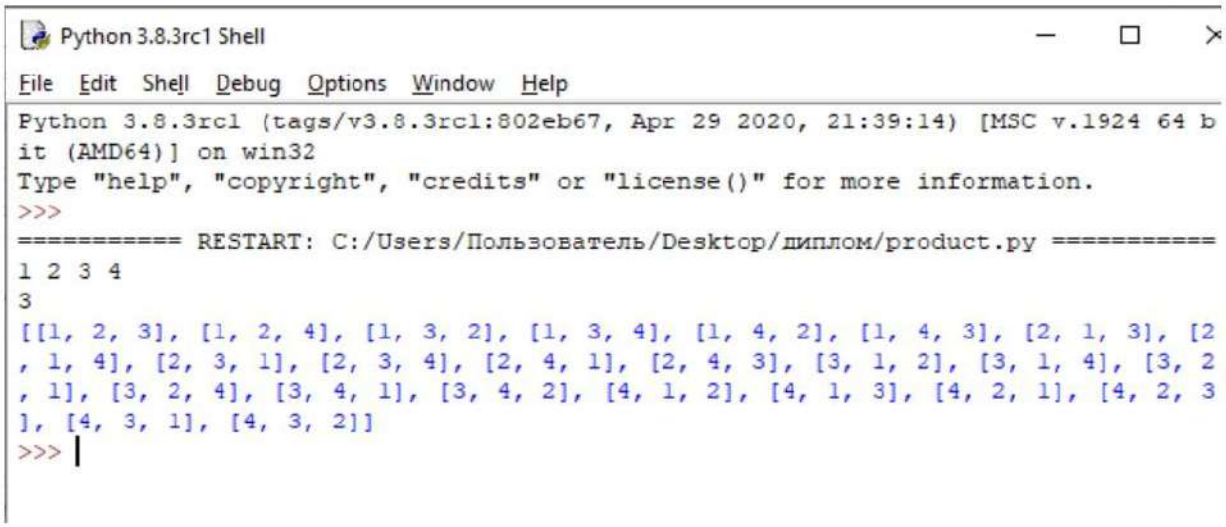

Рис. 1.8. Приклад виконання коду для розміщень

Також є можливість узагальнити та алгоритмізувати задачі на знаходження всіх можливих вибірок з n елементів по m елементів де 1<=m<=n. Одержати всі можливі вибірки можливо реалізувавши алгоритм додавання одиниці в двійковій системі числення до двійкового числа, з кількістю розрядів n. Початковим значенням повинно бути число в двійковій системі числення з кількістю розрядів n:

a...a(n-1)b. Кількість даних вибірок буде знаходитися за формулою: k=2^n. [13 с. 34-35]

Також можна встановити та імпортувати в код спеціальну бібліотеку «itertools». Це значно спростить задачі, в яких єдиною задачею є знаходження представлення перестановок, сполучень, розміщень. Проте майже унеможливить використання частини коду в більш складному. Наприклад, код для знаходження всіх можливих перестановок буде виглядати так (рис. 1.9), сполучень – (рис. 1.10), розміщень – (рис. 1.11). Розміщення від перестановок відрізняються кількістю місць, точніше їх обмеженням, яке вводить користувач.

```
from itertools import *
a[int(i) for i in input().split()]
for i in permutations(a):
    print(i, end=' ')
```

Рис. 1.9. Код для знаходження перестановок з використанням бібліотеки

```
from itertools import *
a[int(i) for i in input().split()]
n=int(input())
for i in combinations(a, n):
    print(i, end=' ')
```

Рис. 1.10. Код для знаходження сполучень з використанням бібліотеки

```
from itertools import *
a[int(i) for i in input().split()]
n=int(input())
for i in permutations(a, n):
    print(i, end=' ')
```

Рис. 1.11. Код для знаходження розміщень з використанням бібліотеки

Переходячи до прикладів використання комбінаторики в алгоритмізації задач з програмування найбільш часто використовується така комбінаторна задача – виведення кількісті елементів, які слід перебрати для отримання рішення залежно від деяких параметрів.

Задача з комбінаторики утворюється у вигляді задачі на підрахунок кількості елементів деякої множини (потужність в математичній термінології). Для розв'язання таких завдань, необхідно мати базові знаннями з теорії множин з розряду властивостей операцій над множинами. Далі задачі можна звести до вираження кінцевої множини через множину(и), потужності яких визначаються за відомими правилами. Для знаходження кількості елементів використовуються правила множення чи додавання, числа сполучень чи розміщень. [24]

Приклад задачі з використанням комбінаторики: Знайти C – кількість n-значних натуральних чисел, які мають у своєму записі хоча б одну цифру 9.

Вхідними даними є натуральне число n ($1 \leq n \leq 10$).

Вихідними даними є невідоме число C.

Аналіз: Для n-значних чисел існує 9 варіантів першої цифри і 10 варіантів інших цифр, або існує $9 \times 10^{n-1}$ всіх таких чисел. Для n-значних чисел, що не містять жодної цифри 9, є 8 варіантів першої цифри і 9 варіантів для всіх інших, виходить є всього $8 \times 9^{n-1}$ шуканих чисел. [9 , с. 142]

Приклад коду мовою Python (рис. 1.12).

```
a = int(input('Введіть число n')) - 1
# вводиться натуральне число n, для збільшення ефективності коду відразу віднімається 1
print(9*10**a-8*9**a)
# обчислення і виведення результату за виведеною раніше формулою
```

Рис. 1.12. Розв'язок задачі

Ця задача має багато різних варіацій. Наприклад, заміна числа, відсутність декількох і більше цифр і т.д.

В олімпіадних задачах чи задачах для підготовки до олімпіади з програмування достатньо задач, які пов'язані лише з комбінаторикою.

Проте є значна кількість складніших задач, де використовуються тільки елементи комбінаторики. В підсумку, елементи комбінаторики активно використовуються в алгоритмізації задач, так знання комбінаторики значно впливає на успіх в розв'язанні задач з програмування. [24]

1.3. Налаштування алгоритму та оцінка ефективності

Олімпіадні змагання посідають особливе місце серед решти організаційних форм навчання.

Метою олімпіад з інформатики є наступне: виявлення талановитих, розвинених та освічених учнів; оцінка загального рівня навчання інформатики в школі; демонстрація найбільшого рівня вимог, встановлення завдань при вивченні інформатики в цілому. [18]

Проте одночасно з'являються додаткові вимоги для вчителя та учня до процесу підготовки до олімпіади з інформатики.

Школяр, має не лише знаходити розв'язок задач, але й визначати оптимальний алгоритм розв'язання завдання, вміти оцінити і проаналізувати ефективність створеного алгоритму.

Вчитель, в свою чергу, зобов'язаний навчити робити все вищезазначене учня самостіно. [23]

В знаходженні розв'язку задачі на олімпіаді з інформатики можливо виділити наступні етапи:

1. Постановка задачі (аналіз безпосередньо умови задачі).
2. Складання плану розв'язку задачі.
3. Побудова математичної моделі та схеми розв'язання задачі.

4. Розробка і реалізація алгоритму.

5. Тестування і налагодження алгоритму.

6. Перевірка розв'язку для запису. [11, с.88]

Розглядаючи безпосередньо створення алгоритму виділяються наступні питання та кроки.

В інформатиці алгоритм – це скінченна і однозначна послідовність однозначно визначених ходів чи дій, які забезпечують розв'язання задачі і для виконання яких потрібен обмежений об'єм оперативної пам'яті і скінченний час. [12; 15]

Алгоритми мають відповідати деяким вимогам:

1. Масовість алгоритму. Алгоритмом має розв'язуватись не тільки конкретна задача, а деякий клас однотипних задач.

2. Скінченність алгоритму. Алгоритм кожного разу мусить завершуватися за скінченну кількість кроків і за скінченний проміжок часу.

3. Однозначність алгоритму (його визначеність чи детермінованість). Всі кроки алгоритму і в цілому весь алгоритм мусить бути однозначно визначений.

Це значить, що всі застосунки алгоритму мають нічим не відрізнятися від попереднього та при введенні одних і тих же даних, результати виконання алгоритму, як окремих кроків, так і всього алгоритму будуть однакові.

4. Правильність алгоритму. Виконання алгоритму з припустимими вхідними даними приводить до отримання необхідного результату. Даний етап є одним із головних етапів створення алгоритму.

5. Ефективність алгоритму. Алгоритм мусить забезпечувати розв'язання задачі в мінімальний проміжок часу та з мінімальними витратами оперативної пам'яті. До того ж повинна оцінюватися ємнісна складність (зростання витрати пам'яті в залежності від розміру вихідних даних). [15, с. 160]

Наступний крок (якщо це допустимо) – це опис задачі термінами деякої формальної моделі. Прикладом може бути наступне. Для обчислювальної задачі

потрібно створити модель на основі основних математичних конструкцій (розв'язування рівняння, системи рівнянь).

Дані дії сприятимуть в побудові близького розв'язку визначеного завдання. До того ж необхідно обрати структуру чи структури вхідних і вихідних даних, через те, що дані дії є не менш важливими.

Наприклад, задачі на впорядкування послідовності елементів мають структуру вхідних та вихідних даних одновимірним масивом, обробка елементів певної таблиці має структуру двовимірного масиву. Обізнаність в більшій кількості структур даних збільшує обсяг задач, які можливо буде розв'язувати. [14, с. 6]

Пошук оптимального алгоритму розв'язання є наступним кроком. Який алгоритм вважається оптимальним? Які існують критерії оптимальності алгоритмів і чи існують вони? Як визначається оптимальність певного алгоритму? [23]

Критерії оптимальності алгоритму можуть бути наступними:

1. Компактність алгоритму. Даний критерій належить до тестового коду програми.

2. Ємнісна складність алгоритму. Передбачає економність використання пам'яті комп'ютера.

3. Часова складність алгоритму. Кінцевий код розв'язання задачі має містити найменш можливу кількість кроків. [14, с. 5-7]

Всі можливі критерії буває нереально задовольнити, через це потрібно знати яким з них варто надавати пріоритет.

Найбільшу увагу потрібно надавати досягненню найменшої кількості дій, і до того ж слід записати алгоритм максимально компактно та зрозуміло і намагатися якомога менше використовувати додаткові змінні.

Визначення оптимальності в певних випадках може робитись використанням визначених формул. Але використання формул в кожному завданні не буде ефективним, буде доцільнішим спочатку спробувати знайти оцінку оптимального

розв'язку аналітично. Для вчителя, який готує учня до олімпіади це значить, що необхідно навчити школяра оцінки оптимальності алгоритмів. [14, с. 7]

Наступним кроком необхідно зауважити час виконання алгоритму. Це означає звернути увагу на дії і на час їх виконання. Вважаються вагомими арифметичні операції і операції порівняння. Кожна операція порівняння вважається еквівалентною за часом іншій. Арифметичні операції в свою чергу ще поділяються ще на дві групи:

- додавання та віднімання, збільшення і зменшення лічильника;
- множення та ділення, отримання залишку по модулю. [23]

Даний поділ викликано тим, що арифметична операція множення виконується довше від додавання. Це означає, що варто надавати перевагу алгоритму, де найменше використовуються операції другої арифметичної групи. [14, с. 7-9]

Проте ефективність алгоритму залежна не лише від використання вищезазначених дій. Таким же вагомим позначається і вибір вхідних даних. Означає, що якщо на вході знаходиться вже відсортована послідовність, отже кількість операцій буде мінімізованою. [23]

Розглядаючи більш детально оцінку і аналіз ефективності алгоритму. Порівняння може відбуватися лише алгоритмів, які розв'язують одну і ту ж саму задачу.

Першочергово критерієм ефективності алгоритму є час виконання алгоритму. Дана оцінка є відносно суб'єктивною, тому, що вона залежна від технічних характеристик комп'ютера, а саме тактової частоти комп'ютера і якості компілятора. Отже варто визначати даний критерій по іншому.

За визначення ефективності алгоритму варто брати не реальну кількість секунд, а приблизну кількість операцій, що виконуються даним алгоритмом. Даний фактор і визначатиме ефективність алгоритму, що, в свою чергу, породжується його складністю. [14, с. 13-19]

Фактором який також варто враховувати є використання додаткової пам'яті. Даний чинник потрібно вважати недоліком.

Далі варто звернути увагу на питання доцільності обчислення числа фактичної кількості операцій алгоритму для вказаних вхідних даних.

Проте це не важливо, через те, що доцільніше розглядати залежність кількості операцій отриманого алгоритму від кількості вхідних даних, якщо детальніше то залежність збільшення кількість операцій від збільшення кількості вхідних даних. Через те, що від зростання обсягу вхідних даних ситуація відносно кількості операцій, що виконуються алгоритмом, порівняно з невеликим їх обсягом, можлива ситуація значного їх зростання. [6]

Вказану характеристику зазвичай вказують як швидкість росту часу роботи алгоритму, або порядок росту часу роботи алгоритму. Даний параметр також обчислюється за допомогою різних формул, які ще необхідно знаходити шляхом аналізу чи віднайти потрібного типу формулу в літературі. [14, с. 13-19]

Через це доцільно буде перейти до методу, яким діти можуть легко користуватися на практиці самостійно. Тобто використовувати метод покрокової деталізації. Даний метод не є засобом перевірки оптимальності чи ефективності алгоритму, але він є основним прийомом створення алгоритмів. Але метод покрокової деталізації можна використовувати і для перевірки роботи алгоритму. [23]

Метод покрокової деталізації – спеціальний прийом, призначений для створення алгоритмів.

Суть даного методу в тому, щоб розбити вихідне завдання на деяку кількість взаємозалежних підзавдань, кожне наступне з яких, також розбивається на підзавдання і т.д. Процес розбиття завершується тоді, коли рішення головного завдання зводиться до виконання деякої кількості простих завдань, для вирішення яких алгоритм складається легко.

Мета методу покрокової деталізації – є деталізація завдань на кожному новому кроці, якщо точніше перехід до не таких загальних та конкретніших завдань на кожному наступному кроці. [12, с. 177]

Використовуючи даний метод можна значно полегшити перевірку правильності і ефективності алгоритму. Зменшення складності полягає в тому, щоб не перевіряти складний алгоритм весь відразу, а перевіряти правильність простих алгоритмів, що є складовими вихідного. Дана перевірка потребує менше зусиль і набагато зменшує шанс впустити помилку.

Наступним кроком є перевірка правильності алгоритму в цілому, спираючись на те, що всі допоміжні алгоритми, які використовуються у вихідному алгоритмі, правильні. Навпаки також буде правильно, припускати правильність всіх допоміжних алгоритмів, перевірити правильність вихідного алгоритму та тільки тоді перевірити правильність всіх допоміжних. [12, с. 182-183]

Переконливим плюсом, щодо використання методу покрокової деталізації є те, що створені в ході методу допоміжні алгоритми є можливим використовувати і при створенні будь-яких інших складних алгоритмів.

Для прикладу, алгоритми для обміну двох елементів, пошуку максимального елемента послідовності – використовуються не в одному типі задач, обмін двох елементів є допоміжним в більшості методів сортування, пошук максимального елемента допоміжний для задач з використання статистичної обробки. [12, с. 183]

На даний момент створено велику кількість алгоритмів, які використовуються як допоміжні, та як самостійні. Розбір та систематичне використання деяких вже створених алгоритмів значно спрощує створення алгоритмів школярем. Підготовка дітей до олімпіади з інформатики, передбачає навчання учнів не одному методу створення алгоритмів і приділення уваги методам перевірки правильності та ефективності алгоритмів. [23]

РОЗДІЛ 2.
ВИВЧЕННЯ ОЛІМПІАДНОГО ПРОГРАМУВАННЯ В НАВЧАЛЬНІЙ РОБОТІ

2.1. Аналіз олімпіадних завдань

Завдання та відповіді олімпіад II-IV турів починаючи з 2015-2016 н.р. та інформацію про олімпіади починаючи з 2012-2013 н.р. можна знайти на сервісі [27]. Інформатика присутня не у всіх роках чи етапах, проте дані матеріали всеодно можна використати для аналізу та підготовки до олімпіади.

Також можна виділити, що майже завжди визначаються обмеження по часу та по пам'яті, і додаються тести, які визначають ефективність алгоритму.

Завдання з елементами комбінаторики зустрічаються в 2016, 2020 років.

Наприклад: задача «Анаграми» 1 туру 2016 року, задача «Щасливі квитки» 2 туру 2016 року, задача «Ланцюжки» 2 туру 2020 року. Розв'язок даних задач повністю базується на елементах комбінаторики.

В першій задачі розв'язок використовується поняття перестановок символів рядків. Якщо один рядок є перестановкою іншого, то виводиться 1, в іншому випадку 0. Попередньо для оптимізації визначається довжина рядків, і якщо вона не рівна, то знаходження потрібної перестановки не виконується.

В задачі «Щасливі квитки» за основу розв'язку береться кількість комбінацій k з n. Доповнена формула (2.1) розв'язку виглядає наступним чином:

$$ans = \sum_{i=1}^{n} \binom{n}{i} \cdot i^{n-i}.$$

Формула 2.1

Потім залишається використати дану формулу в коді обраною мовою програмування.

В задачі «Ланцюжки» також основою розв'язку є базовий елемент комбінаторики, а саме кількість комбінацій з n по m, але з накладанням додаткових умов. Відповіддю буде добуток чотирьох комбінацій з різними параметрами.

В більшості інших задач з олімпіад з інформатики можна зробити розв'язок з використанням частини коду пов'язаного з базовими поняттями комбінаторики. Так як суть олімпіадних задач в знанні теорії з багатьох математичних та інформатичних розділів та вмінні застосувати та поєднати декілька їх на практиці в одному алгоритмі.

Також є два збірники [19], [18] з переліком задач з олімпіадного програмування, з рекомендаціями щодо їх розв'язку та базовим теоретичним матеріалом основ мов програмування Python, C++ та у випадку [7] Pascal.

Розглядаючи перший збірник, тобто [19], в ньому береться за основу Інтернет-ресурс з спортивного програмування з олімпіадними задачами E-Olymp. На даному ресурсі в класифікації «Комбінаторика» зібрано 53 задачі.

Прикладом комбінаторної задачі, розглянутою в збірнику є «7817. Гарне число». Умова задачі: «Гарним» вважається число, що містить лише непарні цифри. Прикладом гарного числа буде 175397, а негарного 8467542. Потрібно знайти скільки існує n-значних гарних чисел. [19, с. 17]

Вхідними даними є ціле невід'ємне числа в межах від 0 до 20 включно, результатом – кількість гарних чисел.

Наприклад, при введені 4 результатом буде 625. Розв'язок повністю базується на елементарних поняттях комбінаторики. На кожен з розрядів числа, яке задовольняє умову задачі, можна поставити будь-яку з п'яти непарних цифр. Отже розв'язком задачі буде 5^n. Код розв'язку задачі мовою Python (рис. 2.1) [19, с. 17]

Варіантом зміни умови даної задачі може бути будь-яка зміна умови «гарності» числа.

```
n=int(input())
print(5**n)
```

Рис. 2.1. Код розв'язку задачі Гарне число

Ще один прикладом задачі з використанням базових понять комбінаторики є «1355. Кількість n-значних чисел, що містить 7». Умова задачі: Знайти кількість – k n-значних натуральних чисел, які містять у своєму запису як мінімум одну цифру 7. [19, с. 18]

Вхідними даними є натуральне число n в межах від 1 до 10 включно, результатом є кількість n-значних натуральних чисел, що задовольняють умову задачі.

Прикладом виконання коду може бути: Вхідні дані: 2, вихідні дані: 18. Відповідь знаходиться наступними міркуваннями кількість всіх n-значних натуральних чисел рівна $9 \cdot 10^{n-1}$, кількість n-значних чисел, що не містять семірки рівна $8 \cdot 9^{n-1}$, так як на кожне місце зменшується кількість претендентів на 1. [19, с. 18]

Щоб отримати розв'язок задачі тепер достатньо відняти від першого результату другий. Код розв'язку задачі мовою Python (рис. 2.2)

```
n=int(input())
k=9*10**(n-1)-8*9**(n-1)
print(k)
```

Рис. 2.2. Код розв'язку задачі Гарне число

Варіантом зміни умови даної задачі може бути будь-яка зміна цифри чи кількості цифр.

Наступним прикладом задачі з збірника, в якій використовуються елементи комбінаторики є «7460. Поїздка на екскурсію».

В даній задачі потрібно за відомої кількості хлопців – n та дівчат – m розподілити їх по кімнатах з k місцями, з умовою, що хлопчики та дівчата не можуть бути в одній кімнаті. [19, с. 19]

Вхідними даними є цілі числа n – кількість хлопців, m – кількість дівчат, k – кількість місць в кімнаті.

У збірнику в прикладі розв'язання не використовуються елементи комбінаторики. Але в розв'язанні можна використати сполучення, не просто їх формулу, а, зважаючи на певні обмеження в умові, створити формулу з сполученнями (формула 2.2), відповідно до розв'язування комбінаторних задач з обмеженнями.

Приклад коду розв'язку зі збірника [19, с. 20] (рис. 2.3).

$$C_{n+m-k+1}^{k}$$

Формула 2.2

```
File  Edit  Format  Run  Options  Window  Help
n, m, k = map(int, input().split())
p = n//k+int(n%k>0)
q = m/k+int(m%k>0)
print(p+q)
```

Рис. 2.3. Код зі збірника

Наступна задача із цього ж збірника не розв'язується лише базовими поняттями комбінаторики, а можна використати в розв'язанні їх як деяку основу коду. Назва задачі – «2392. Цікава сума».

Її умовою є знаходження суми найбільшого та найменшого трицифрових чисел, що можуть бути утворені з заданого натурального числа n способом

перестановки цифр. Вхідними даними даної задачі є саме натуральне число n, яке належить проміжку від 100 до 999 включно. [19, с. 30]

Вихідними даними задачі 2392 є сума знайдених найбільшого і найменшого числа, що задовольняють умову.

Для розв'язання представленої задачі потрібно модифікувати код, зазначений вище, для перестановок, не просто для їх виводу, а для обрахування потрібної формули.

Аналізуючи другий збірник, а саме [7], в даному посібнику розглядаються авторські обласні, районі та інтернет олімпіади, пропонуються варіанти їх розв'язання мовою програмування Pascal та пояснення ходу самого розв'язання.

Прикладом задачі з використанням елементів комбінаторики є задача «N-значні числа».

Умова задачі: Знайти кількість n-значних натуральних чисел, в яких сума цифр дорівнює певному числу m, яке знаходиться в діапазоні від 1 до 900, n, в свою чергу в діапазоні від 1 до 100 включно. [7, с. 49]

Очевидно, що при n рівному одиниці можливі значення суми чисел, що задовольнятимуть умову будуть належати діапазону від 1 до 9 включно і буде по одному відповідно до кожної суми. Відповідно формула (формула 2.3) для розв'язку буде наступною.

$$A[n, m] = A[n - 1, m] + A[n, m - 1]$$

Формула 2.3

Стосовно випадку коли значення суми чисел строго більше 10, тоді результат зменшується наступним чином A[n-1, m-10]. [7, с. 49]

В даному збірнику також є задачі в яких можна використати елементи комбінаторики, як допоміжні частини коду.

Першим прикладом таких задач можна зазначити задачу з назвою «Симетрія чисел». В даній задачі умовою є знаходження числа n-значних паліндромів, тобто числа, яке однаково читається в обох напрямках. [7, с. 15]

В посібнику дана задача розв'язується довгим кодом зі створенням спеціальної функції. Мовою програмування Python дана задача розв'язується в декілька рядків (рис. 2.4). Приклад виконання (рис. 2.5).

```
s = input()
s1 = 'YES' if s == s[::-1] else 'NO'
print(s1)
```

Рис. 2.4. Код розв'язку задачі Симетрія чисел

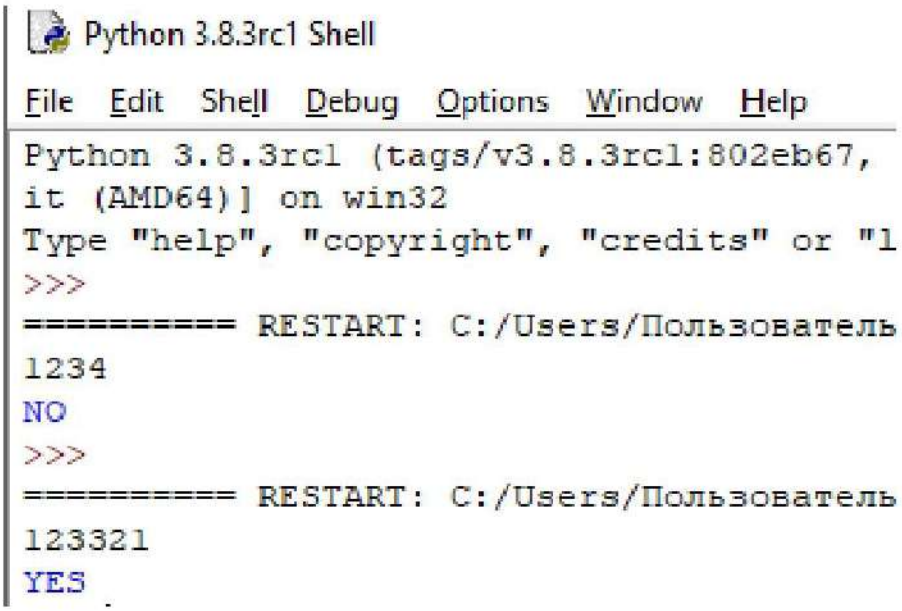

Рис. 2.5. Приклад виконання коду

Наступним прикладом є задача «Олімпіадна задача про олімпіаду». Дана задача схожа на задачу з попереднього посібника з назвою «Поїздка на екскурсію», але дещо з іншими умовами.

Суть даної задачі полягає в тому, щоб розподілити a[i] учасників з n команд на c комп'ютерів в k кабінетах, k – потрібно знайти, з умови, що в одному кабінеті не можуть знаходитися представники будь-якої одної і тієї ж команди. [7, с. 23]

В посібнику задача розв'язується діями з масивами, а саме сумою елементів масиву, діленням та знаходженням максимального елементу. Однак дану задачу можна розв'язати за допомогою комбінаторики.

Також далі пропонується схожа задача, до якої вже не додається через це розв'язку.

Умовою задачі «Інтернет-кафе» є знаходження мінімального часу надсилання листів компанією із n друзів з c комп'ютерів з інтернет-кафе (на надсилання одного листа витрачається одна хвилина). [7, с. 25]

Схожу на дві попередні, задачу буде подано в прикладах задач для підготовки та їх розв'язання тому далі буде проаналізовано три приклади комбінаторних задач, розміщених на електронному сервісі E-olymp та створено їх розв'язки.

Першою задачею на розгляд є задача «17. Садівник-художник».

Умовою даної задачі є визначення кількості способів розфарбування n дерев (n знаходиться в діапазоні від 1 до 50 включно), якщо два однакові кольори не можуть бути поруч, кольорів всього три: білий, синій та помаранчевий.

Вхідними даними задачі є кількість дерев, вихідними даними є кількість способів зафарбування. [2]

Розв'язком даної задачі буде (рис. 2.6), де c – кількість кольорів фарби, n – кількість дерев, значення яких вводиться з клавіатури.

```
n = int(input())
c = 3
for i in range(n-1):
    c=c*2
print(c)
```

Рис. 2.6. Код розв'язку задачі

Тести даного розв’язку на E-olymp (рис. 2.7). Тести з даного сайту показують, що в майбутньому потрібно ще оптимізувати даний алгоритм, так як він використовує більше пам’яті ніж 75% розв’язків, що вносились на ресурс.

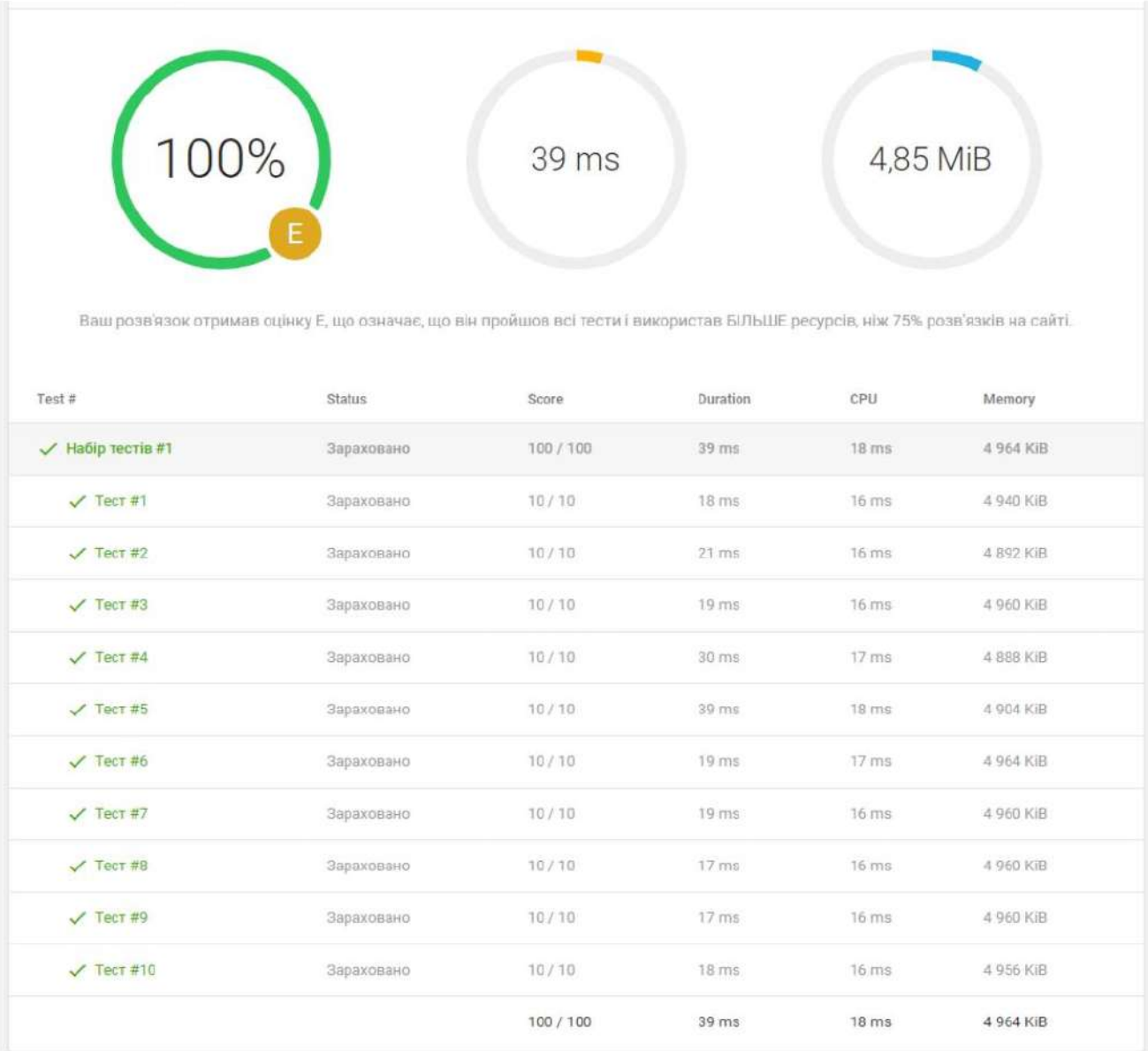

Рис. 2.7. Тести розв'язку задачі

Наступною задачею на аналіз є «843. Перестановки (2)».

Умова даної задачі полягає в виведенні всіх перестановок символів введеного рядка довжиною m, що лежить в діапазоні від 2 до 8 включно. [2]

Тут очевидне пряме використання перестановок, проте варто врахувати слова з повторенням літер, тобто робити перевірку на повтори перед тим, як використовувати до масиву функцію перестановок. Кількість перестановок мовою програмування Python знаходиться наступним чином (рис 2.8).

```
import math
n = int(input())
print(math.factorial(n))
```

Рис. 2.8. Кількість перестановок

Також в наступній задачі «2385. Кількість перестановок» також є очевидним та прямим використання перестановок, але вже підрахунок їх кількості.

2.2. Сервіси для підготовки до олімпіад з інформатики

При підготовці до олімпіад з інформатики з'являється необхідність використання значної кількості різних ресурсів для вичерпної підготовки до змагання.

Змістовним джерелом, що полегшить підготовку буде вагома кількість Інтернет-ресурсів, використовуючи які можна результативно вивчати необхідний матеріал і здобувати потрібні навички розв'язання задач з програмування. [4]

На даний момент Інтернет-ресурси безпосередньо використовуються в організації дистанційних олімпіад.

Інтернет-ресурси та їх використання дають можливість учням приєднуватися до online-змагань, що схожі за типами та за структурою до тих, які

використовуються в предметних олімпіадах. Подібні тренування дадуть можливість школярам бути більш підготовленими до олімпіад. [4]

Для економії часу під час перевірки розв'язків учнів, які готуються до олімпіади, доцільно в комп'ютерному класі мати власну систему перевірки задач, або розв'язувати задачі з сайтів із системами автоматичної перевірки розв'язків (http://acm.timus.ru/, http://www.ttb.by, http://www.acm.lviv.ua, http://www.e-olimp.com.ua, www.olymp.vinnica.ua, http://codeforces.ru).

Використовуючи дані сайти, можна прискорити та оптимізувати роботу підготовки до олімпіади, виходячи з того, що:

- на сайтах є великий набір задач;
- розв'язки задач можна відправити на перевірку і за лічені секунди отримати результат;
- рівні умови перевірки, усувається людський фактор оцінювання. [11, с. 62-64]

На сайті http://e-olimp.com.ua є можливість створювати групи учасників. У створених групах можна проводити змагання на базі існуючих задач, які доступні лише членам даної групи, проводити обговорення задач у межах створеної групи, переглядати рейтинг учасників даної групи.

Створений Інтернет-портал www.e-olimp.com дає можливість полегшити роботу учителя, тренера під час підготовки до олімпіади з інформатики, відкриває можливості обдарованим учням самостійно працювати, розвиватися, обмінюватися досвідом з однодумцями з різних куточків країни та світу. [11, с. 63]

На даному Інтернет- порталі можна знаходиться величезна кількість задач, проте їх можна сортувати за складністю та класифікації і відповідно легше знайти потрібну задачу чи групу задач.

Окрім цього вчитель може використовувати інші платформи для створення своїх матеріалів, завдань та інших розробок для підготовки учнів до олімпіад з програмування.

Наприклад, є можливість створити свій курс на MS Moodle чи створити окремий клас для учнів, що готуються в Google Classroom, чи створити свій сайт для підготовки з використанням будь-якої безкоштовної платформи, наприклад, Google Sites. Можливостей безліч.

2.3. Приклади задач для підготовки та їх розв'язання

При виборі мови програмування для навчання учнів інформатиці та при їх підготовці до олімпіад, з огляду на кількість методичної літератури з підготовки до олімпіад та загальної кількості літератури по мові програмування, з'являються думки про вибір мови програмування Pascal. Проте більшість учнів та вчителів вважають дану мову вимираючою і все менше учасників олімпіад її використовують.

Розглядаючи дозволені мови програмування на олімпіадах з інформатики наступним претендентом на розгляд є мова C++.

Порівнюючи кількість методичної літератури та посібників та збірників з підготовки до олімпіад їх кількість з використанням мови програмування C++ значно менша, проте є вагома кількість Інтернет-ресурсів з підготовки даною мовою.

Для вибору з доступних на олімпіаді мов залишилось розглянути Python. . Стислий опис та приклади розв'язків даною мовою подані в [17].

Детальніший розгляд теорії, основні функції та методи мови програмування Python в [3].

Проте, навіть беручи до уваги значну кількість літератури, і Інтернет-ресурсів, що допомагають отримати базові знання з Python, даною мовою програмування в Україні мізерна кількість літератури з підготовки до олімпіад, так як на олімпіадах досить своєрідні алгоритми і шукаючи їх на просторах всесвітньої мережі можна так нічого й не знайти. Певною мірою пояснення алгоритмів мовою

програмування Python надається в якості розборів олімпіадних задач на Інтернет-ресурсі [1]. В підготовці до олімпіад з інформатики також доцільно застосовувати і посібник [16]. [28]

Перевагами мови програмування Python для олімпіадного програмування можна назвати:

- Відсутність потреби підключення окремим кодом «довгих чисел» та довгих рядків, так як вони вбудовані (якщо переглядати приклади кодів, які пов'язані з даними величинами в мережі Інтернет, то можна побачити суттєве спрощення коду мовою Python в порівнянні з іншими мовами програмування);
- Розв'язання задач з використанням комбінаторики мовою програмування Python значно простіше, відповідні коди навіть з цієї роботи іншими мовами прописувались би набагато довше;
- Робота з масивами значно простіша;
- Коди виконання алгоритмів мовою програмування Python набагато компактніші. [28]

Проте для олімпіадного програмування в мови Python є деякий недолік – час виконання, навіть значно довші коди мовою програмування C++ іноді виконуються швидше.

Зважаючи на всі переваги та недоліки мов надалі приклади розв'язання задач та використання їх на уроці будуть представлені мовою програмування Python.

Для порівняння код для перестановок мовою програмування C++ буде виглядати наступним чином (рис. 2.9).

```
const int N = 3;
std::set<int> IntSet;
std::set<int> IntSetTmp;
int A[N];
void InitSet()
{
     for(int i = 1; i <= N; i++)
     {
          IntSet.insert( i );
     }
}
void Per(std::set<int> S, int K)
{
     int i;
     std::set<int>::iterator it = S.begin();
     if(S.empty())
     {
          for(i=0;i<N;i++)
               printf("%d ", A[i]);
           printf("\n");
     }
     else
     {
          for( i=1; i <= N; i++ )
               if(S.find(i) != S.end())
               {
                    A[K] = i;
                    IntSetTmp = S;
                    IntSetTmp.erase(i);
                    Per(IntSetTmp, K+1);
               }
     }
}
void main()
{
   InitSet();
   Per(IntSet,0);
}
```

Рис. 2.9. Перестановки C++

Перелік умов задач для підготовки з їх розв'язками:

1. Скільки існує прямокутників, виміри яких набувають значень від m до n?

Вхідні дані: два цілі числа, виміри прямокутників.

Вихідні дані: кількість існуючих прямокутників.

Приклад вхідних даних: 2, 7.

Приклад вихідних даних: 21.

Дана задача є досить простою, для її розв'язку достатньо створити код для знаходження кількості перестановок з повтореннями з n-1 по m.

2. N хлопців та m дівчат грають у волейбол. Скількома способами їх можна розділити на дві команди по s осіб, якщо у кожній команді має бути принаймні k дівчат? Команди мають назви.

   Вхідні дані: чотири цілих числа: n, m, k, s.

   Вихідні дані: кількість способів поділу.

   Приклад вхідних даних: 5, 3, 4.

   Приклад вихідних даних: 30.

   Ця задача також не викликає труднощів, якщо знати базові елементи комбінаторики. Її розв'язком буде добуток сполучень (формула 2.4)

$$C_m^k \cdot C_n^{s-k}$$

Формула 2.4

3. Скільки існує n-значних натуральних чисел?

   Вхідні дані: ціле число n.

   Вихідні дані: кількість n-значних натуральних чисел.

   Приклад вхідних даних: 3.

   Приклад вихідних даних: 900.

   Задача розв'язується простими міркуваннями: кількість таких чесл для одиниці буде 9, для двійки – 90, для трійки – 900 і т.д. І шляхом підбору формули для прикладів отримується формула з розв'язку задачі (рис. 2.10).

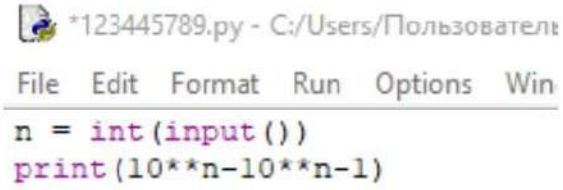

Рис. 2.10. Розв'язок задачі 3

4. Скільки різних n-цифрових натуральних чисел можна скласти з цифр a[i], якщо цифри різні?

   Вхідні дані: ціле число n, список a[i].

   Вихідні дані: кількість n-цифрових натуральних чисел.

   Приклад вхідних даних: 4, [0, 1, 2, 4, 5].

   Приклад вихідних даних: 120.

   Розв'язком даної задачі є кількість розміщень з кількісті елементів у списку (len(a)) по n (рис. 2.11).

```
import math
n = int(input())
a = [int(i) for i in input().split()]
c = math.factorial(len(a))/math.factorial(len(a)-n)
print(c)
```

Рис. 2.11. Розв'язок задачі 4

5. Вивести числа з попередньої задачі на екран.

   Розв'язком даної задачі є виведення всіх розміщень на екран, що є вищезазначеним в першому розділі.

6. Вивести всі можливі впорядкування літер введеного слова.

Розв'язком цієї задачі є виведення всіх можливих перестановок літер введеного слова. Код розв'язку для чисел та пояснення перероблення для літер є в Розділі 1.

7. Скількома способами з n учнів класу можна вибрати старосту, заступника та куторга?

   Вхідні дані: ціле число n.

   Вихідні дані: кількість способів вибору.

   Приклад вхідних даних: 25.

   Приклад вихідних даних: 2300.

   Розв'язком даної задачі є число сполучень з n по 3.

8. Скількома способами можна записати рядок з n нулів та k одиниць, щоб жодні дві одиниці не стояли поруч.

   Вхідні дані: цілі числа n, k.

   Вихідні дані: кількість способів запису.

   Приклад вхідних даних: 6, 9.

   Приклад вихідних даних: 84.

   Розв'язком даної задачі є наступна формула (формула 2.5) запис схожих мовою програмування Python вже вказувався вище. Розв'язок здійснювався наступним чином: розташовуємо нулі з проміжками, тоді місць для одиниць n-1+2 = n+1 (в проміжках та по краях), тоді вибрати місця для одиниць можна в (формула 2.5) спосіб.

$$C_{n+1}^{m}$$

Формула 2.5

9. Дано n-значне натуральне число, вивести цифри даного числа в зворотному порядку.

   Вхідні дані: ціле число n (запис якого – abcd).

Вихідні дані: зворотне число (запис якого – dcba).

Приклад вхідних даних: 1234.

Приклад вихідних даних: 4321.

Дану задачу можна розв'язати двома способами: перший – перестановки в лексикографічному порядку, виводимо останню; другий – ділення націло та остача від ділення націло числа на 10.

10. На яку кількість частин n кіл розбивають площину?

Вхідні дані: ціле число n.

Вихідні дані: кількість частин.

Приклад вхідних даних: 3.

Приклад вихідних даних: 8.

Відповідь на дану задачу це код на (рис. 2.12).

Кількість кіл визначається кількістю точок перетину з іншими колами на площині, тобто розбиттям їх на частини.

```
n = int(input())
print(n**2-n+2)
```

Рис. 2.12. Розв'язок задачі 10

2.4. Використання на уроках інформатики

На сьогоднішній день змістові лінії алгоритми та програмування вивчаються в наступних класах та з наступною приблизною кількістю годин:

- В 5-му класі назва змістової лінії «Алгоритми та програми» відводиться не менше 40 % часу на опанування теми, тобто не менше 14 годин. [22]

- В 6-му класі вивчаються «Алгоритми та програми», кількість відведеного часу аналогічна часу в п'ятому класі. [22]
- В 7-му класі змістова лінія, що вивчається, має назву «Алгоритми та програми» кількість годин, які мають приділятися на опанування даної змістової лінії, така ж як і в попередніх. [22]
- У 8-му класі на вивчення тем змістової лінії «Алгоритми та програми» відводиться ті ж 40 % часу, однак це вже 28 годин. [22]
- В 9-му класі змістова лінія називається «Алгоритми та програми», на опанування її тем має бути не менше 30 % – 21 година. [22]
- В 10-му класі профільного рівня вивчається «Мова програмування та структури даних». [21]
- В 10-11-х класах рівня стандарт – «Креативне програмування», що є вибірковим модулем на вивчення якого відводиться 35 годин. [21]
- В 11-му класі профільного рівня вивчається «Алгоритми. Парадигми та технології програмування». [21]

Час в усіх попередніх прикладах вказується приблизним через те, що потрібний для досягнення поставлених результатів час, визначається вчителем самостійно залежно від рівня попередньої підготовки школярів, вибраної методики навчання, наявного обладнання школи тощо.

Далі представлено три конспекти уроків для вибіркового модулю «Креативне програмування» для 10-11-х класах рівня стандарт.

“___” _____ 202__ року

Урок №__. Правила написання ефективного та читабельного коду. Перевірка роботи алгоритму.

Мета: сформувати в учнів нові знання про правила написання ефективного та читабельного коду. Забезпечити зв’язок з минулим досвідом. Розвивати логічне мислення. Формувати наступні вміння: актуалізувати, аналізувати, зрівнювати, виокремлювати основне. Формувати такі компетентності як інформаційну, навчально-пізнавальну, комунікативну, соціально-трудову та компетентність особистісного самовдосконалення. Забезпечити диференційований підхід.

Тип уроку: засвоєння нових знань.

Обладнання: підручники, зошити, комп’ютери, проектор.

Програмне забезпечення: Python.

Хід уроку

І. Організаційний етап

- ***Привітання з класом***

  Доброго дня, учні.

- ***Повідомлення теми і мети уроку***

На даному уроці ми вивчимо правила написання ефективного та читабельного коду. Навчимось перевіряти роботу алгоритму.

Нагадування про правила техніки безпеки та поведінки в кабінеті інформатики.

ІІ. Мотивація навчальної діяльності

Створення програмного коду є творчістю, але окрім фантазії, знань як створити код, необхідно, щоб його змогли прочитати інші. А ми його змогли перевірити та оптимізувати – зробити ще швидшим, коротшим, з меншою витратою пам’яті. Про це сьогодні і поговоримо.

ІІІ. Актуалізація опорних знань

Перевірка домашнього завдання з минулого уроку. Опитування з попереднього матеріалу.

Вивчення нового матеріалу

Розповідь вчителя про назви змінних та функцій, коментарі.

Вказування вимог до алгоритмів з 1.3. Критерії оптимальності та шляхи їх досягнення з 1.3. Та пояснення роботи методу покрокової деталізації з 1.3. Демонстрація оптимізації та перевірки роботи одного з алгоритмів для задач 3, 5, 6, 9, 10.

Фізкультхвилинка

Усвідомлення набутих знань та формування вмінь і навичок

Використання задач 3, 5, 6, 9, 10 на вибір учнів. Створення розв'язку для однієї та аналіз створеного алгоритму.

Релаксація

Підведення підсумків уроку

1. Як зробити код читабельним?
2. Які є вимоги до алгоритмів?
3. Який алгоритм можна назвати оптимізованим?
4. Яка суть методу покрокової деталізації?
5. Краще швидкість роботи алгоритму чи його менший обсяг?

Домашнє завдання

Опрацювати конспект. Проаналізувати алгоритм з попередніх уроків на ефективність та при можливості оптимізувати його.

“___” _____ 202__ року

Урок №__. Модульність.

Мета: дати визначення модульності. Забезпечити зв’язок з минулим досвідом. Розвивати логічне мислення. Формувати наступні вміння: актуалізувати, аналізувати, зрівнювати, виокремлювати основне. Формувати такі компетентності як інформаційну, навчально-пізнавальну, комунікативну, соціально-трудову та компетентність особистісного самовдосконалення. Забезпечити диференційований підхід.

Тип уроку: засвоєння нових знань;

Обладнання: комп’ютери, підручники, зошити, проектор.

Програмне забезпечення: Python.

Хід уроку

І. Організаційний етап

- ***Привітання з класом***
- ***Повідомлення теми і мети уроку***

На сьогоднішньому уроці ми вивчимо поняття модульності.

Нагадування про правила техніки безпеки та поведінки в кабінеті інформатики.

ІІ. Мотивація навчальної діяльності

При створенні алгоритмів ми можемо використовувати модулі, це значно спростить нашу роботу, лише потрібно запам’ятати потрібні модулі та їх функції.

ІІІ. Актуалізація опорних знань

Перевірка домашнього завдання з минулого уроку. Опитування з попереднього матеріалу.

Вивчення нового матеріалу

Ознайомлення учнів з поняттям модульності та основними модулями мови програмування Python. Ознайомлення з модулем math та іншими, їх основними

функціями та з модуля math функціями math.comb та math.factorial. Демонстрація розв'язку однієї з задач 1, 2, 4, 7, 8, з використанням модуля math.

Фізкультхвилинка

Усвідомлення набутих знань та формування вмінь і навичок

Розв'язування задач 1, 2, 4, 7, 8 дві, три на вибір учнів.

Релаксація

Підведення підсумків уроку

1. Що таке модуль?
2. Які є приклади модулів в мові програмування Python?
3. Назвіть приклади функцій модуля math.

Домашнє завдання

Опрацювати конспект.

“___” _____ 202__ року

Урок №__. Рекурсія.

Мета: дати визначення рекурсії. Забезпечити зв’язок з минулим досвідом. Розвивати логічне мислення. Формувати наступні вміння: актуалізувати, аналізувати, зрівнювати, виокремлювати основне. Формувати такі компетентності як інформаційну, навчально-пізнавальну, комунікативну, соціально-трудову та компетентність особистісного самовдосконалення. Забезпечити диференційований підхід.

Тип уроку: засвоєння нових знань;

Обладнання: комп’ютери, підручники, зошити, проектор.

Програмне забезпечення: Python.

Хід уроку

І. Організаційний етап

- ***Привітання з класом***
- ***Повідомлення теми і мети уроку***

На сьогоднішньому уроці ми вивчимо поняття рекурсії.

Нагадування про правила техніки безпеки та поведінки в кабінеті інформатики.

ІІ. Мотивація навчальної діяльності

При створенні алгоритмів ми можемо використовувати модулі, це значно спростить нашу роботу, лише потрібно запам’ятати потрібні модулі та їх функції.

ІІІ. Актуалізація опорних знань

Перевірка домашнього завдання з минулого уроку. Опитування з попереднього матеріалу.

Вивчення нового матеріалу

Ознайомлення учнів з поняттям рекурсії як функції в мові програмування Python та як виклику підпрограми з неї самої. Означення умови використання

рекурсії, показ прикладів використання та прикладів використання в інших галузях. Показ знаходження факторіалу числа за допомогою рекурсії.

Фізкультхвилинка

Усвідомлення набутих знань та формування вмінь і навичок

Створити алгоритм розв'язання задачі: Для вказаних n, m, k знайти $n^m\%k$.

Релаксація

Підведення підсумків уроку

1. Що таке рекурсія з точки зору програмування?
2. Що таке факторіал числа?
3. Яка є головна умова для її використання?
4. Назвіть приклади рекурсій.
5. Наведіть приклади рекурсій в інших галузях.

Домашнє завдання

Опрацювати конспект.

ВИСНОВКИ

У роботі вказано особливості підготовки до олімпіад з програмування з використанням комбінаторних задач та мови програмування Python. Проведене дослідження дозволяє зробити наступні висновки.

1. Ґрунтуючись на аналізі науково-педагогічних джерел, ми охарактеризували підготовку до олімпіади з програмування, визначили основні моменти, які варто враховувати. Зокрема олімпіади з інформатики проводяться в чотири етапи. Учні мають вивчати не тільки мову і методи програмування, а ще необхідно врахувати, що результат залежить і від підготовки школярів до стресової ситуації, їх вмінні концентруватися на завданні та його виконанні і розподілу часу на нього, знаходження нестандартних розв'язків завдань. Варто також звертати увагу на вже розроблені схеми, методи та підходи до підготовки до олімпіад і не забувати про поділ завдань олімпіад на теми.

2. Означено основні поняття комбінаторики: перестановки, сполучення та розміщення, створено коди мовою програмування Python в іншому випадку описано розв'язок для реалізації знаходження їх кількості чи представлення їх всіх можливих у коді.

3. Уточнено особливості оцінки ефективності та налаштування алгоритму. Зокрема вказано та розглянуто критерії ефективності алгоритму: компактність, ємнісна складність та часова складність. Та розглянуто метод покрокової деталізації для налаштування алгоритму.

4. Здійснено аналіз завдань з олімпіад з інформатики минулих років на наявність задач з використанням елементів комбінаторики, для цього використано офіційний сайт СОІППО, збірники задач з програмування та Інтернет-сервіс E-olymp.

5. Здійснено аналіз сервісів для підготовки до олімпіад з інформатики. Зокрема, вказано http://acm.timus.ru/, http://www.ttb.by, http://www.acm.lviv.ua,

http://www.e-olimp.com.ua, www.olymp.vinnica.ua, http://codeforces.ru та вказано про можливість створення ресурсу особисто вчителя.

6. За результатами дослідження було розроблено авторські задачі та їх розв’язки для підготовки до олімпіад з програмування з теми комбінаторика.

7. Здійснено аналіз навчальних програм відносно наявності тем програмування та алгоритми. Також розроблено авторські конспекти уроків, де вказано використання методів розв’язання задач з використанням базових елементів комбінаторики, розроблених задач та розглянутих методів.

Дане дослідження не вичерпує всіх питань, які пов’язані з особливостями розв’язування комбінаторних задач при підготовці до олімпіад з інформатики. Перспективи даного дослідження полягають в удосконаленні розроблених алгоритмів, методів, створенні нових завдань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Atamanyuk S., Semenikhina O., Shyshenko I. Theoretical fundamentals of innovation of higher education in Ukraine. Pedagogy and Education Management Review (PEMR). Tallinn, Estonia, 2021. Issue 2(4). P. 30-36.

2. Codeforces. URL: http://codeforces.com/

3. Dehtiarova N., Petrenko S., Rudenko Yu. Pedagogical design in the context of blended learning for future computer science teachers. Modern approaches to the development of knowledge management. Ljubljana. Slovenia. pp. 313-323.

4. Drushlyak M. G., Semenikhina O. V., Kondratiuk S. M., Krivosheya T. M., Vertel A. V., Pavlushchenko N. M. The Automated Control of Students Achievements by Using Paper Clicker Plickers. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 688-692.

5. Drushlyak M. G., Shishenko I. V., Borozenets N. S., Nekyslykh K. M., Semenikhina O. V. Computer Probabilistic Models Construction and Analysis of Professional Activity of their Use by Ukrainian Mathematics Teachers. Proceedings of 44 International convention on information and communication technology, electronics and microelectronics “MIPRO 2021”, Opatija (Croatia), 28 September – 1 October, 2021. P. 712-717. DOI: 10.23919/MIPRO52101.2021.9596868

6. Drushlyak M., Semenikhina O., Proshkin V., Sapozhnykov S. Training pre-service mathematics teacher to use mnemonic techniques. Journal of Physics: Conference Series. 1840 (2021), 012006. C.1-12 DOI:10.1088/1742-6596/1840/1/012006

7. E-olymp. URL: https://www.eolymp.com/uk/

8. Kudrina, O., Shpileva, V., Klius, Y., Lavrova, O., Esmanov, O., & Semenikhina, O. Industrial enterprise tax transaction costs planning using digital tools. TEM Journal. 2020. Volume 9(2), P. 619-624. DOI:10.18421/TEM92-26

9. Lazorenko S. A., Semenikhina O. V. Development of Information and Digital Culture of Future Specialists in Physical Culture and Sports as a Modern Problem

of Education. Science and Education a New Dimension. Pedagogy and Psychology, VIII (95), Issue 239, 2020 Nov. P. 29-32.

10. Lutz M. Learning Python, Fifth Edition. Cambridge: O'Reilly, 2013. 1540 p.

11. Okhrimenko O., Semenikhina O., Shyshenko I. Future teachers' readiness for the digital modernization of inclusive education. New challenges in the development of future specialists: collective monograph. Universitatea Dunarea de Jos Galati, Romania, 2021. P. 83-94.

12. Okhrimenko O., Semenikhina O., Shyshenko I. Readiness of future teachers for digital modernization of inclusive education. Innovative Approaches to Ensuring the Quality of Education, Scientific Research and Technological Processes : collective monograph. 2021. No 3.6.15. P. 694-700.

13. Omelyanenko, V., Kudrina, O., Semenikhina, O., Zihunov, V., Danilova, O. & Liskovetska, T. Conceptual aspects of modern innovation policy. European Journal of Sustainable Development. 2020. Volume 9 (2). P. 238-249. DOI:10.14207/ejsd.2020.v9n2p238

14. Ostroha M., Drushlyak M., Shyshenko I., Naboka O., Proshkin V., Semenikhina O. On the use of social networks in teachers' career guidance activities. Smyrnova-Trybulska E. (ed.). (2021) E-learning in COVID-19 Pandemic Time. "E-learning" Series. Vol. 13 (2021) (Pp. 113-124) Katowice-Cieszyn: Studio Noa for University of Silesia.

15. Petrenko S., Dehtiarova N. Increasing teachers' ict-competency level in the after-graduate education process. Інноваційна педагогіка. Вип. 21. Т. 3. 2020. С. 73-77.

16. Rudenko Yu., Rozumenko A., Kryvosheya T., Karpenko O., Semenikhina O. Online Training during the COVID-19 Pandemic: Analysis of Opinions of Practicing Teachers in Ukraine Proceedings of 44 International convention on information and communication technology, electronics and microelectronics "MIPRO 2021", Opatija (Croatia), 28 September – 1 October, 2021. DOI: 10.23919/MIPRO52101.2021.9596799

17. Rudenko Yu., Semenikhina O. Analysis of distance learning experience in colleges of Sumy region of Ukraine. Education during a pandemic crisis: problems and prospects / Eds. Tetyana Nestorenko & Tadeusz Pokusa Opole, 2020. P. 175-181

18. Rudenko Yuliia, Olha Naboka, Larysa Korolova, Khana Kozhukhova, Olena Kazakevych, Olena Semenikhina. Online Learning With the Eyes of Teachers and Students in Educational Institutions of Ukraine. TEM Journal. Volume 10, Issue 2, P. 922-931. DOI: 10.18421/TEM102-55.

19. Semenikhina O. et al. The Formation of Skills to Visualize by the Tools of Computer Visualization. TEM Journal. 2020. Volume 9(4). P. 1704-1710. DOI: 10.18421/TEM94-51

20. Semenikhina O. V. The Using Interactive Methods In The Formation Of Conflictological Culture Of Specialist. International Scientific Journal «Future Science: Youth Innovations Digest». 2019. Volume 3, Issue 3. P. 44-48

21. Semenikhina O., Drushlyak M., Lynnyk S., Kharchenko I., Kyryliuk H., Honcharenko O. On Computer Support of the Course “Fundamentals of Microelectronics” by Specialized Software: the Results of the Pedagogical Experiment. TEM Journal. 2020. Volume 9 (1). P. 309-316. DOI: 10.18421/TEM91-43

22. Semenikhina O., Drushlyak M., Yurchenko A., Udovychenko O., Budyanskiy D. The use of virtual physics laboratories in professional training: the analysis of the academic achievements dynamics. ICT in Research, Education and Industrial Applications (ICTERI-2020) : 16th International Conference. October, 06-10, 2020. Kharkiv. P. 423-429.

23. Semenikhina O., Proshkin V., Drushlyak M. Mathematical knowledge control automation within dynamic mathematics programs. E-learning and STEM Education / Scientific Editor Eugenia Smyrnova-Trybulska. Katowice–Cieszyn, 2019. P. 571-586. .

24. Semenikhina O., Proshkin V., Naboka O. Application of Computer Mathematical Tools in University Training of Computer Science and Mathematics Pre-

service Teachers. International Journal of Research in E-Learning, 2020, 6(2), 1-23. https://doi.org/10.31261/IJREL.2020.6.2.06

25. Semenikhina O., Yurchenko A., Sbruieva A., Kuzminskyi A., Kuchai O., Bida O. The Open Digital Educational Resources In IT-Technologies: Quantity Analysis. Information technologies and learning tools. V. 75. Issue 1. P. 331-348 https://doi.org/10.33407/itlt.v75i1.3114

26. Semenikhina Olena V., Proshkin Volodymyr V. The main problems of using computer mathematical tools in university education. Інформаційні технології в освіті та науці: Збірник наукових праць. Випуск 12. Мелітополь: ФОП Однорог Т.В., 2021. 204 с. С.9-11.

27. Semenikhina, O., Yurchenko, A., Udovychenko, O., Petruk, V., Borozenets, N., Nekyslykh, K. Formation Of Skills To Visualize Of Future Physics Teacher: Results Of The Pedagogical Experiment. Revista Romaneasca Pentru Educatie Multidimensionala, 2021, 13(2), 476-497. https://doi.org/10.18662/rrem/13.2/432

28. Semenog O., Semenikhina O., Oleshko P., Prima R., Varava O., Pykaliuk R. Formation of Media Educational Skills of a Future Teacher in the Professional Training. Revista Românească pentru Educaţie Multidimensională. 2020. Volume 12. Issue 3, P. 219-245. https://doi.org/10.18662/rrem/12.3/319.

29. Shamonia, V. H., Semenikhina, O. V., Proshkin, V. V., Lebid, O. V., Kharchenko, S. Y., & Lytvyn, O. S. Using the proteus virtual environment to train future IT professionals. CEUR Workshop Proceedings, 2547. P. 24-36.

30. Shishenko I. V., Shamonia V. H., Loboda V. S., Punko V. V., Khvorostina Yu. V. and Voitenko A. A. Studying dynamic mathematics software in the professional training of teachers of computer science, mathematics, and IT specialists. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 683-687.

31. Shkolnyi, O., & Tykhonenko, Y. (2021). USING OF ICT DURING PREPARATION FOR EIA: WAYS TO SEARCH FOR THE OPTIMAL MODEL. *Physical and Mathematical Education*, *30*(4), 13–19. https://doi.org/10.31110/2413-1571-2021-030-4-002

32. Udovychenko O., Chkana Ya., Yurchenko A., Khvorostina Yu. Introduction of didactic games in the educational process. Фізико-математична освіта. 2019. Вип. 4(22). Частина 2. URL: https://fmo-journal.fizmatsspu.sumy.ua/publ/8-1-0-621.

33. Udovychenko, O. M., Ostroha, M. M., Chernysh, A. E., Kudrina, O. Y., Bondarenko, Y. A., & Kurienkova, A. V. (2020). The use of electronic textbooks in the learning process: A statistical analysis. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 608-611. doi:10.23919/MIPRO48935.2020.9245146

34. Voitenko A., Semenikhina O. To the question about inclusive educational space in the training of informatics of children with intellectual disabilities. Education. Innovation. Practice. 2019. Issue 2 (6). P. 6-9.

35. Yurchenko A., Drushlyak M., Sapozhnykov S., Teplytska S., Koroliova L., Semenikhina O. Using online IT-industry courses in the computer sciences specialists' training. International Journal of Computer Science and Network Security. Vol. 21 No. 11 pp. 97-104. http://paper.ijcsns.org/07_book/202111/20211113.pdf

36. Yurchenko A., Semenikhina O., Rudenko Yu., Shamonia V. The Digital Technology in IT-Education: the View of Ukrainian University. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. №4 (482). С. 129-133. https://doi.org/10.15589/znp2020.4(482).15

37. Yurchenko A., Shamonia V., Udovychenko O., Momot R., Semenikhina O. Improvement of Teacher Qualification in the Field of Computer Animation: Training or Master Class? Proceedings of 44 International convention on information and communication technology, electronics and microelectronics "MIPRO 2021", Opatija

(Croatia), 28 September – 1 October, 2021. P. 683-687. DOI: 10.23919/MIPRO52101.2021.9596946

38. Yurchenko A.O., Udovychenko O.M., Rozumenko A.M., Chkana Y.O., Ostroha M.M. (2019). Regional Computer Graphics Competition as a Tool of Influence on the Profession Choice: Experience of Sumy Region of Ukraine. 42nd International Convention on Computers in Education (MIPRO) (May 20 – 24, 2019), Opatija, Croatia, 2019, pp. 909-914.

39. Абрамик М.В., Лещук С.О., Олексюк В.П. Використання хмарних технологій у процесі навчання майбутніх учителів інформатики основам програмування. Фізико-математична освіта. 2018. Випуск 4(18). С. 7-11.

40. Атаманюк С.І., Шишенко І.В., Семеніхіна О.В. Інновації в освіті та специфічні принципи підготовки майбутніх фахівців їх використовувати. Фізико-математична освіта. Суми, 2020. Вип. 4(26). Ч. 2. С. 13-16.

41. Бобровицька С.Ф., Семеніхіна О.В. Стан розробленості проблеми підготовки майбутніх учителів початкової школи до застосування електронних освітніх ресурсів у професійній діяльності. Педагогіка та психологія. 2019. Вип. 62. С. 23-29.

42. Будянський Д.В., Друшляк М.Г., Семеніхіна О.В., Харченко І.В., Горбачук В.О., Чашечникова О.С. Типологія електронних ресурсів у формуванні риторичної культури фахівця. Інформаційні технології і засоби навчання. 2021. 81(1), С. 82-96. https://doi.org/10.33407/itlt.v81i1.4292

43. Вакал Ю.С., Шамоня В.Г. Організація педагогічного експерименту із використанням сучасних інформаційних технологій: навч. посіб. Суми: СумДПУ імені А. С. Макаренка, 2020. 156 с.

44. Ворожбит А.В., Рибак О.С. Огляд курсу за вибором «основи верстки та веб-програмування». Фізико-математична освіта. 2018. Випуск 1(15). С. 20-27

45. Гобунов М. О., Франчук Н. П. Використання Інтернет-ресурсів під час підготовки учнів до олімпіад з інформатики. 2021.

46. Горошко Ю. В., Міца О. В., Мельник В. І. Методичні підходи до розв’язування олімпіадних задач з інформатики. Інформаційні технології і засоби навчання, 2019, Том 71, №3. С. 40-52.

47. Данильчук О.М., Діденко М.М. Чи потрібна математика в програмуванні? InterConf, (35). 2020.

48. Дегтярьова Н., Петренко С. Актуальні питання формування цифрових компетентностей вчителів різних дисциплін під час підвищення кваліфікації. Актуальні питання гуманітарних наук: міжвузівський збірник наукових праць молодих вчених Дрогобицького державного педагогічного університету імені Івана Франка. Дрогобич: Видавничий дім «Гельветика», 2020. Вип. 27. Том 2. С. 167-170.

49. Дегтярьова Н.В., Петренко С.І. Змішане навчання як чинник формування навичок самоосвіти у майбутніх вчителів інформатики. Вісник Вінницького політехнічного інституту. 2(143). 2019. С. 117-122.

50. Дегтярьова Н.В., Руденко Ю.О., Вернидуб Г.О. Формування вміння у майбутніх учителів працювати над науковим текстом. Педагогіка формування творчої особистості у вищій і загальноосвітній школах: зб. наук. праць. Запоріжжя: КПУ, 2020. Вип. 68. Т.1. С. 240-243.

51. Дегтярьова Н.В., Руденко Ю.О., Шамоня В. Г., Семеніхіна О.В. Методика вирішення нечітких багатокритеріальних задач вибору варіантів. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. № 3 (481). С. 124-128. https://doi.org/10.15589/znp2020.3(481).16

52. Дідковський В. Л., Матвійчук С. В. Олімпіади з інформатики. Харків. 2012. С. 236.

53. Дідковський В. Л., Матвійчук С. В. Олімпіади з інфороматики. Основа. 2012. С. 236.

54. Друшляк М. Г., Юрченко А. О., Розуменко А. М., Розуменко А. О., Семеніхіна О. В. Ефективні форми підвищення кваліфікації вчителів у галузі

комп’ютерної анімації. Відкрите освітнє е-середовище сучасного університету, 2021, 10 (1), С. 77-88. https://doi.org/10.28925/2414-0325.2021.108

55. Жданова Ю.Д., Спасітєлєва С.О., Шевченко С.М. Формування у студентів ІТ-спеціальностей компетентностей в області захисту інформації з використанням криптографічних служб .NET FRAMEWORK Фізико-математична освіта. 2019. Випуск 1(19). С. 48-54.

56. Жмурко О. І., Охріменко Т. О. Олімпіади з програмування. Прості задачі. Умань Візаві 2020. – 301 с.

57. Жуковський С. С. Аналіз, дослідження та розв’язування конкурсних задач під час учнівської олімпіади з інформатики. Вісник Житомирського державного університету. Випуск 53. Педагогічні науки. 2010. С. 153-157.

58. Жуковський С. С. Педагогічні умови підготовки обдарованих школярів до олімпіад з інформатики. Київ. 2013. – 235 с.

59. Зарецька І. Т. Інформатика. 10-11 класи. Частина 2. Київ. – 2006. – 289 с.

60. Караванова Т. П. Інформатика. Методи побудови алгоритмів та їх аналіз. Обчислювальні алгоритми. К.: Генеза, 2009. – 212 с.

61. Караванова Т. П. Інформатика. Методи побудови алгоритмів та їх аналіз. Необчислювальні алгоритми. К.: Генеза, 2007. – 212 с.

62. Кнут Д. Э. Искусство программирования. Том 1. Основные алгоритмы = The Art of Computer Programming. Volume 1. Fundamental Algorithms / под ред. С. Г. Тригуб (гл. 1), Ю. Г. Гордиенко (гл. 2) и И. В. Красикова (разд. 2.5 и 2.6). – 3. – Москва: Вильямс, 2002. – Т. 1. – 720 с.

63. Кобильник, Т., Когут, У., & Жидик, В. (2021). МЕТОДИЧНІ АСПЕКТИ ВИВЧЕННЯ ОСНОВ АЛГОРИТМІЗАЦІЇ І ПРОГРАМУВАННЯ МОВОЮ PYTHON У ШКІЛЬНОМУ КУРСІ ІНФОРМАТИКИ У СТАРШИХ КЛАСАХ. *Фізико-математична освіта, 31*(5), 36–44. https://doi.org/10.31110/2413-1571-2021-031-5-006

64. Кормен Т. Х., Лейзерсон И. И., Ривест Р. Л., К. Штайн. Алгоритмы: построение и анализ, 3-е изд. Москва. «Вильямс». 2013. С. 1328.

65. Креневич А. П. Python у прикладах і задачах. Частина 1. Структурне програмування Навчальний посібник із дисципліни "Інформатика та програмування". Київ. ВПЦ "Київський Університет". 2017. С. 206.

66. Кривонос О. М. Учнівські олімпіади з інформатики (сучасний етап). 2008.

67. Кудін, А. (2021). ОРГАНІЗАЦІЯ САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ НА БАЗІ СИМУЛЯЦІЙНОГО ЛАБОРАТОРНОГО ПРАКТИКУМУ З ОСНОВ ЦИФРОВОЇ ЕЛЕКТРОНІКИ. *Фізико-математична освіта*, *30*(4), 61–67. https://doi.org/10.31110/2413-1571-2021-030-4-009

68. Кузьменко А.В. Огляд навчальних програм з інформатики для учнів старших класів загальноосвітнього навчального закладу. Фізико-математична освіта. 2017. Випуск 3(13). С. 93-99.

69. Мартиненко О., Чкана Я., Удовиченко О. Управління самостійною роботою майбутніх учителів математики у віртуальному навчальному середовищі через використання електронної версії робочого зошиту. Педагогічні науки: теорія, історія, інноваційні технології. 2020. № 2 (96). С. 144-153.

70. Матвійчук С. В., Жуковський С. С. Практикум програмування Python/C++ на e-olymp.com. Житомир. 2019. С. 231.

71. Мельник М. С., Маланюк Н. Б. Методика розв’язування олімпіадних задач з програмування у шкільному курсі інформатики. Сучасні інформаційні технології та інноваційні методики навчання: досвід, тенденції, перспективи № 1. 2017. С. 183-186.

72. Мітельман І.М. Навчання розв’язування олімпіадних задач, пов’язаних із цілою частиною дійсного числа, за допомогою властивостей точок розриву кусково-сталих функцій. Фізико-математична освіта. 2019. Випуск 2(20). С. 107-113

73. Навчальні програми для 10-11 класів https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-programi/navchalni-programi-dlya-10-11-klasiv

74. Навчальні програми для 5-9 класів. https://mon.gov.ua/ua/osvita/zagalna-serednya-osvita/navchalni-programi/navchalni-programi-5-9-klas

75. Носаченко Д. С., Шамоня В. Г. До питання з підготовки до олімпіади з інформатики. Матеріали результатів досліджень молодих науковців. Випуск 15. Том 1. Суми. 2021. С. 53-56

76. Носаченко Д. С., Шамоня В. Г. Елементи комбінаторики в алгоритмічних задачах. Матеріали результатів досліджень молодих науковців. Випуск 15. Том 2. Суми. 2021. С. 29-30.

77. Одінцова О.О. Особливості створення математичних моделей задач, що вивчаються в лінійному програмуванні. Фізико-математична освіта. 2016. Випуск 1(7). С. 105-113.

78. Острога М.М., Шамоня В.Г. Модель формирования готовности будущих бакалавров среднего образования к использованию цифровых технологий в профориентационной деятельности. Science and Education a New Dimension. Pedagogy and Psychology, IX (97), Issue: 246, 2021. P.25-28.

79. Павленко Л.В., Павленко М.П., Хоменко В.Г., Хоменко С.В., Скурська М.М. Інноваційні підходи до вивчення статистики майбутніми ІТ-фахівцями на основі використання мови програмування R. Фізико-математична освіта. 2020. Випуск 1(23). С. 97-105.

80. Петренко С., Петренко Л. Модель формування інформатичної компетентності майбутніх учителів інформатики в процесі фахової підготовки. Педагогічні науки: теорія, історія, інноваційні технології. Суми: СумДПУ імені А. С. Макаренка, 2020. № 2 (96) С. 154-164. DOI 10.24139/2312-5993/2020.02/154-164

81. Петренко С., Петренко Л. Формування готовності майбутніх учителів інформатики до професійної діяльності. Педагогічні науки: теорія, історія,

інноваційні технології. Суми: СумДПУ імені А. С. Макаренка, 2019. № 10 (94). С. 95-105. DOI 10.24139/2312-5993/2019.10/095-106.

82. Петренко С.І. Аналіз проблеми безпечної роботи учнів початкових класів у мережі Інтернет // Петренко С.І. / Вісник університету імені Альфреда Нобеля. Серія «Педагогіка і психологія». Педагогічні науки. 2020. № 1 (19) С. 85-92. DOI: 10.32342/2522-4115-2020-1-19-9

83. Петренко С.І., Дегтярьова Н.В. Формування ІКТ-компетентності викладачів на курсах підвищення кваліфікації. Наукові записки Серія: Педагогічні науки Випуск 186 - Кропивницький: РВВ ЦДПУ ім. В. Винниченка, 2020. с. 150-155.

84. Про затвердження Положення про Всеукраїнські учнівські олімпіади, турніри, конкурси з навчальних предметів, конкурси-захисти науково-дослідницьких робіт, олімпіади зі спеціальних дисциплін та конкурси фахової майстерності. URL: https://zakon.rada.gov.ua/laws/show/z1318-11#Text

85. Прошкін В., Хоружа Л., Семеніхіна О. Теорія і практика професійної підготовки майбутніх учителів математики та інформатики засобами цифрових технологій. Теоретичні та практичні аспекти використання математичних методів та інформаційних технологій в освіті й науці: моногр. / за заг. ред. О. Литвин. К.: Київ. ун-т ім. Б. Грінченка, 2021. 332 с. С.48-74.

86. Руденко Ю. О., Дегтярьова Н. В., Юрченко А. О., Семеніхіна О. В. Використання елементів нечіткої логіки у гуманітарних дослідженнях. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. № 1 (479). С. 130-134. https://doi.org/10.15589/znp2020.1(479).17

87. Руденко Ю.О., Дегтярьова Н.В. Електронні ресурси та сервіси інтернет в контексті реалізації електронного навчання. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С.56-86.

88. Руденко Ю.О., Лобова В.В. З досвіду проведення олімпіад з інформатики серед студентів коледжів. Фізико-математична освіта. 2019. Випуск 1(19). С. 184-188.

89. Семеніхіна О. В., Прошкін В. В., Друшляк М. Г. Використання прийомів мнемотехніки в процесі навчання математики. Математика в рідній школі. 2020. №5 (219). С. 2-7.

90. Семеніхіна О., Юрченко А. Професійна підготовка фахівця: організація онлайн-опитування для визначення потреб у зміні освітньої програми. Освіта. Інноватика. Практика. 2019. Issue 2(6). P. 36-43.

91. Семеніхіна О., Юрченко А., Удовиченко О. Формування умінь візуалізувати начальний матеріал у майбутніх учителів фізики: результати педагогічного експерименту. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 99-117.

92. Семеніхіна О.В., Бобровицька С.Ф. Особливості практичної підготовки вчителів до використання ЕОР у початковій школі. Фізико-математична освіта. 2020. Вип. 1(23). Частина 2. С. 72-77.

93. Семеніхіна О.В., Юрченко А.О., Удовиченко О.М. Формування умінь візуалізувати начальний матеріал у майбутніх учителів фізики: результати педагогічного експерименту. Фізико-математична освіта. 2020. Вип. 1(23). С. 122-128.

94. Семеног О., Семеніхіна О. Медіаосвітні уміння майбутнього вчителя та особливості їх формування у процесі професійної підготовки. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С.118-140.

95. Сенчевський В. О. Перші кроки в теорію ймовірностей. Харків «Основа». 2008. с. 42-65

96. Сумський обласний інститут післядипломної освіти. Головна. Сторінка методиста. Конкурси, олімпіади (архів). URL:

http://www.soippo.edu.ua/index.php/34-2010-11-24-15-07-23/78-2010-11-24-16-53-32?Itemid=16 дата звернення: 10.10.2021

97. Удовиченко О.М. Критерії та показники рівнів готовності майбутніх учителів інформатики до професійної діяльності. Вісник Черкаського національного університету. Серія «Педагогічні науки». Черкаси, 2020. Вип. 2.2020. С. 142-147.

98. Харченко В. М. До питання використання мови python в олімпіадах з інформатики. Вісник навчально-наукового інституту точних наук і економіки Збірник наукових праць. Ніжин. 2018. С. 97-101.

99. Харченко І.І., Удовиченко О.М. Результати експериментального формування культури професійної комунікації майбутніх фахівців з економіки. Вісник Черкаського національного університету. Серія «Педагогічні науки». Черкаси, 2020. Вип. 1.2020. С. 146-150.

100. Хворостіна Ю.В., Удовиченко О.М., Юрченко А.О. Особливості використання дидактичних ігор на уроках математики. Інноваційна педагогіка. 2019. Вип. 19. Том 3. С. 141-146. https://doi.org/10.32843/2663-6085-2019-19-3-29

101. Чередник І.В., Руденко Ю.О., Семеніхіна О.В. Труднощі навчання учнів системам числення і кодуванню інформації та шляхи їх запобігання. Фізико-математична освіта. 2020. Випуск 2(24). Частина 2. С. 21-27.

102. Шамоня В., Семеніхіна О. Комп’ютерна візуалізація роботи логічних елементів інформаційної системи на базі PROTEUS. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 87-98.

103. Шамшина Н.В. Методичні аспекти вивчення СУБД ACCESS: створення інформаційних систем. Професійна підготовка вчителя в умовах цифрового освітнього середовища / за заг. ред. О.В. Семеніхіної. Суми, 2020. С. 140-178.

104. Юрченко А.О., Семеніхіна О.В., Хворостіна Ю.В., Удовиченко О.М., Петренко С.І. Навчання програмувати в старшій школі крізь призму чинних навчальних програм. Фізико-математична освіта. 2019. Випуск 2(20). Ч. 2. С. 48-55.

105. Юрченко А.О., Семеніхіна О.В., Хворостіна Ю.В., Удовиченко О.М., Петренко С.І. Навчання програмувати в старшій школі крізь призму чинних навчальних програм. Фізико-математична освіта. 2019. Вип. 2(20). Ч. 2. С. 48-55. DOI 10.31110/2413-1571-2019-022-4-021.

106. Юрченко А.О., Удовиченко О.М., Хворостіна Ю.В., Петренко С.І. Дослідження рівня знань майбутніх учителів фізики при використанні цифрових лабораторій. Фізико-математична освіта. 2019. Вип. 4(22). С. 137-141. DOI 10.31110/2413-1571-2019-022-4-021.

107. Atamanyuk S., Semenikhina O., Shyshenko I. Theoretical fundamentals of innovation of higher education in Ukraine. Pedagogy and Education Management Review (PEMR). Tallinn, Estonia, 2021. Issue 2(4). P. 30-36.

108. Codeforces. URL: http://codeforces.com/

109. Dehtiarova N., Petrenko S., Rudenko Yu. Pedagogical design in the context of blended learning for future computer science teachers. Modern approaches to the development of knowledge management. Ljubljana. Slovenia. pp. 313-323.

110. Drushlyak M. G., Semenikhina O. V., Kondratiuk S. M., Krivosheya T. M., Vertel A. V., Pavlushchenko N. M. The Automated Control of Students Achievements by Using Paper Clicker Plickers. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 688-692.

111. Drushlyak M. G., Shishenko I. V., Borozenets N. S., Nekyslykh K. M., Semenikhina O. V. Computer Probabilistic Models Construction and Analysis of Professional Activity of their Use by Ukrainian Mathematics Teachers. Proceedings of 44 International convention on information and communication technology, electronics and

microelectronics “MIPRO 2021”, Opatija (Croatia), 28 September – 1 October, 2021. P. 712-717. DOI: 10.23919/MIPRO52101.2021.9596868

112. Drushlyak M., Semenikhina O., Proshkin V., Sapozhnykov S. Training pre-service mathematics teacher to use mnemonic techniques. Journal of Physics: Conference Series. 1840 (2021), 012006. C.1-12 DOI:10.1088/1742-6596/1840/1/012006

113. E-olymp. URL: https://www.eolymp.com/uk/

114. Kudrina, O., Shpileva, V., Klius, Y., Lavrova, O., Esmanov, O., & Semenikhina, O. Industrial enterprise tax transaction costs planning using digital tools. TEM Journal. 2020. Volume 9(2), P. 619-624. DOI:10.18421/TEM92-26

115. Lazorenko S. A., Semenikhina O. V. Development of Information and Digital Culture of Future Specialists in Physical Culture and Sports as a Modern Problem of Education. Science and Education a New Dimension. Pedagogy and Psychology, VIII (95), Issue 239, 2020 Nov. P. 29-32.

116. Lutz M. Learning Python, Fifth Edition. Cambridge: O’Reilly, 2013. 1540 p.

117. Okhrimenko O., Semenikhina O., Shyshenko I. Future teachers’ readiness for the digital modernization of inclusive education. New challenges in the development of future specialists: collective monograph. Universitatea Dunarea de Jos Galati, Romania, 2021. P. 83-94.

118. Okhrimenko O., Semenikhina O., Shyshenko I. Readiness of future teachers for digital modernization of inclusive education. Innovative Approaches to Ensuring the Quality of Education, Scientific Research and Technological Processes : collective monograph. 2021. No 3.6.15. P. 694-700.

119. Omelyanenko, V., Kudrina, O., Semenikhina, O., Zihunov, V., Danilova, O. & Liskovetska, T. Conceptual aspects of modern innovation policy. European Journal of Sustainable Development. 2020. Volume 9 (2). P. 238-249. DOI:10.14207/ejsd.2020.v9n2p238

120. Ostroha M., Drushlyak M., Shyshenko I., Naboka O., Proshkin V., Semenikhina O. On the use of social networks in teachers’ career guidance activities.

Smyrnova-Trybulska E. (ed.). (2021) E-learning in COVID-19 Pandemic Time. "E-learning" Series. Vol. 13 (2021) (Pp. 113-124) Katowice-Cieszyn: Studio Noa for University of Silesia.

121. Petrenko S., Dehtiarova N. Increasing teachers' ict-competency level in the after-graduate education process. Інноваційна педагогіка. Вип. 21. Т. 3. 2020. С. 73-77.

122. Rudenko Yu., Rozumenko A., Kryvosheya T., Karpenko O., Semenikhina O. Online Training during the COVID-19 Pandemic: Analysis of Opinions of Practicing Teachers in Ukraine Proceedings of 44 International convention on information and communication technology, electronics and microelectronics "MIPRO 2021", Opatija (Croatia), 28 September – 1 October, 2021. DOI: 10.23919/MIPRO52101.2021.9596799

123. Rudenko Yu., Semenikhina O. Analysis of distance learning experience in colleges of Sumy region of Ukraine. Education during a pandemic crisis: problems and prospects / Eds. Tetyana Nestorenko & Tadeusz Pokusa Opole, 2020. P. 175-181

124. Rudenko Yuliia, Olha Naboka, Larysa Korolova, Khana Kozhukhova, Olena Kazakevych, Olena Semenikhina. Online Learning With the Eyes of Teachers and Students in Educational Institutions of Ukraine. TEM Journal. Volume 10, Issue 2, P. 922-931. DOI: 10.18421/TEM102-55.

125. Semenikhina O. et al. The Formation of Skills to Visualize by the Tools of Computer Visualization. TEM Journal. 2020. Volume 9(4). P. 1704-1710. DOI: 10.18421/TEM94-51

126. Semenikhina O. V. The Using Interactive Methods In The Formation Of Conflictological Culture Of Specialist. International Scientific Journal «Future Science: Youth Innovations Digest». 2019. Volume 3, Issue 3. P. 44-48

127. Semenikhina O., Drushlyak M., Lynnyk S., Kharchenko I., Kyryliuk H., Honcharenko O. On Computer Support of the Course "Fundamentals of Microelectronics" by Specialized Software: the Results of the Pedagogical Experiment. TEM Journal. 2020. Volume 9 (1). P. 309-316. DOI: 10.18421/TEM91-43

128. Semenikhina O., Drushlyak M., Yurchenko A., Udovychenko O., Budyanskiy D. The use of virtual physics laboratories in professional training: the analysis of the academic achievements dynamics. ICT in Research, Education and Industrial Applications (ICTERI-2020) : 16th International Conference. October, 06-10, 2020. Kharkiv. P. 423-429.

129. Semenikhina O., Proshkin V., Drushlyak M. Mathematical knowledge control automation within dynamic mathematics programs. E-learning and STEM Education / Scientific Editor Eugenia Smyrnova-Trybulska. Katowice–Cieszyn, 2019. P. 571-586. .

130. Semenikhina O., Proshkin V., Naboka O. Application of Computer Mathematical Tools in University Training of Computer Science and Mathematics Pre-service Teachers. International Journal of Research in E-Learning, 2020, 6(2), 1-23. https://doi.org/10.31261/IJREL.2020.6.2.06

131. Semenikhina O., Yurchenko A., Sbruieva A., Kuzminskyi A., Kuchai O., Bida O. The Open Digital Educational Resources In IT-Technologies: Quantity Analysis. Information technologies and learning tools. V. 75. Issue 1. P. 331-348 https://doi.org/10.33407/itlt.v75i1.3114

132. Semenikhina Olena V., Proshkin Volodymyr V. The main problems of using computer mathematical tools in university education. Інформаційні технології в освіті та науці: Збірник наукових праць. Випуск 12. Мелітополь: ФОП Однорог Т.В., 2021. 204 с. С.9-11.

133. Semenikhina, O., Yurchenko, A., Udovychenko, O., Petruk, V., Borozenets, N., Nekyslykh, K. Formation Of Skills To Visualize Of Future Physics Teacher: Results Of The Pedagogical Experiment. Revista Romaneasca Pentru Educatie Multidimensionala, 2021, 13(2), 476-497. https://doi.org/10.18662/rrem/13.2/432

134. Semenog O., Semenikhina O., Oleshko P., Prima R., Varava O., Pykaliuk R. Formation of Media Educational Skills of a Future Teacher in the Professional Training.

Revista Românească pentru Educaţie Multidimensională. 2020. Volume 12. Issue 3, P. 219-245. https://doi.org/10.18662/rrem/12.3/319.

135. Shamonia, V. H., Semenikhina, O. V., Proshkin, V. V., Lebid, O. V., Kharchenko, S. Y., & Lytvyn, O. S. Using the proteus virtual environment to train future IT professionals. CEUR Workshop Proceedings, 2547. P. 24-36.

136. Shishenko I. V., Shamonia V. H., Loboda V. S., Punko V. V., Khvorostina Yu. V. and Voitenko A. A. Studying dynamic mathematics software in the professional training of teachers of computer science, mathematics, and IT specialists. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 683-687.

137. Shkolnyi, O., & Tykhonenko, Y. (2021). USING OF ICT DURING PREPARATION FOR EIA: WAYS TO SEARCH FOR THE OPTIMAL MODEL. *Physical and Mathematical Education*, *30*(4), 13–19. https://doi.org/10.31110/2413-1571-2021-030-4-002

138. Udovychenko O., Chkana Ya., Yurchenko A., Khvorostina Yu. Introduction of didactic games in the educational process. Фізико-математична освіта. 2019. Вип. 4(22). Частина 2. URL: https://fmo-journal.fizmatsspu.sumy.ua/publ/8-1-0-621.

139. Udovychenko, O. M., Ostroha, M. M., Chernysh, A. E., Kudrina, O. Y., Bondarenko, Y. A., & Kurienkova, A. V. (2020). The use of electronic textbooks in the learning process: A statistical analysis. MIPRO 2020 : Proceedings of 43 International convention on information and communication technology, electronics and microelectronics, 28 вересня – 2 жовтня 2020, Opatija (Croatia). 2020. P. 608-611. doi:10.23919/MIPRO48935.2020.9245146

140. Voitenko A., Semenikhina O. To the question about inclusive educational space in the training of informatics of children with intellectual disabilities. Education. Innovation. Practice. 2019. Issue 2 (6). P. 6-9.

141. Yurchenko A., Drushlyak M., Sapozhnykov S., Teplytska S., Koroliova L., Semenikhina O. Using online IT-industry courses in the computer sciences specialists' training. International Journal of Computer Science and Network Security. Vol. 21 No. 11 pp. 97-104. http://paper.ijcsns.org/07_book/202111/20211113.pdf

142. Yurchenko A., Semenikhina O., Rudenko Yu., Shamonia V. The Digital Technology in IT-Education: the View of Ukrainian University. Збірник наукових праць Національного університету кораблебудування імені адмірала Макарова, 2020. №4 (482). С. 129-133. https://doi.org/10.15589/znp2020.4(482).15

143. Yurchenko A., Shamonia V., Udovychenko O., Momot R., Semenikhina O. Improvement of Teacher Qualification in the Field of Computer Animation: Training or Master Class? Proceedings of 44 International convention on information and communication technology, electronics and microelectronics "MIPRO 2021", Opatija (Croatia), 28 September – 1 October, 2021. P. 683-687. DOI: 10.23919/MIPRO52101.2021.9596946

144. Yurchenko A.O., Udovychenko O.M., Rozumenko A.M., Chkana Y.O., Ostroha M.M. (2019). Regional Computer Graphics Competition as a Tool of Influence on the Profession Choice: Experience of Sumy Region of Ukraine. 42nd International Convention on Computers in Education (MIPRO) (May 20 – 24, 2019), Opatija, Croatia, 2019, pp. 909-914.