**День 01. Основи програмування мовою Python**

**Рівень А**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | **Умова:**  Напишіть програму, яка обчислює індекс маси тіла (ІМТ) для особи. Спочатку додаток має зчитувати зріст та масу користувача. Для обчислення ІМТ використовується одна з 2 формул. Якщо зріст вимірюється в дюймах, а маса – у фунтах, ІМТ визначається так:  Для метрів та кілограмів ІМТ виражається відношенням  Виведіть відповідне значення ІМТ; вони будуть аналізуватись у наступних задачах.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  veight = int(input("Маса: "))  height = int(input("Зріст: "))  yes = str(input("Поставте \"+\",якщо вага в кг, а ріст в м, інакше поставте \"-\": "))  if yes == "+":  imt = veight / height \*\* 2  elif yes == "-":  imt = 703 \* veight / height \*\* 2  else:  print("Неправильнй символ!")  print(imt)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 2. | **Умова:**  Створіть програму, яка зчитує тиск у кПа (кілопаскалях). Після зчитування даних, програма повинна повідомити про еквівалентний тиск у фт/дм2 (фунтах на квадратний дюйм), міліметрах ртутного стовпчика та атмосферах.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  kPa = int(input("Тиск(кПа): "))  psi = kPa \* 145.038 \* 10\*\*(-3)  mmrt = kPa \* 7.5  tex\_atm = kPa \* 0.102 \* 10\*\*(-1)  phis\_atm = kPa \* 0.987 \* 10\*\*(-2)  print(  "Міліметри ртутного стовпчика -",mmrt,"\n"+  "Фунти на квадратний дюйм -",psi,"\n"+  "Технічна атмосфера -",tex\_atm,"\n"+  "Фізична атмосфера -",phis\_atm)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 0,5 бала |
| 3. | Умова:  Створіть програму, яка зчитує букву англійського алфавіту від користувача. Якщо користувач ввів a, e, i, o або u, програма повинна вивести повідомлення, що введена буква – голосна. Якщо користувач вводить y, програма має повідомити, що інколи y – голосна, а інколи - приголосна. Інакше слід вивести повідомлення, що буква приголосна.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  import random  bukwa = (input("Буква:")).lower()  if bukwa in "aouei":  print("Голосна")  elif bukwa == "y":  rand = random.randint(0,1)  if rand == 1:  print("Голосна")  else:  print("Приголосна")  else:  print("Приголосна")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 4. | Умова:  Програма в цій вправі повинна починати роботу зі зчитування вартості їжі, замовленої з ресторану (назва страви, кількість порцій, ціна за порцію). Потім програма обчислить податок та чайові за їжу. Розраховуйте на чайові в розмірі 14% від замовлення та податок у розмірі 18% від замовлення. Вивід вашої програми має включати розмір податку, чайових та загальну суму для оплати. Формат виводу передбачає представлення всіх значень з використанням 2 знаків після коми.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  print("Для зупинення введіть \"end\"")  gen = 0  while True:  name = input("Назва страви: ")  amount = int(input("Кількість страви: "))  price = int(input("Ціна страви: "))  gen += amount\*price  dali = input("Ще страава:")  if dali == "end":  break  chayovi = gen\*0.14  podatok = gen \* 0.18  sum\_price = gen + podatok + chayovi  print("Ціна: ",gen,"\n"+  "Чайоі: %.2f" % chayovi,"\n"+  "Податок: %.2f" % podatok,"\n"+  "Всього: ", sum\_price)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 5. | Умова:  Напишіть програму, яка спочатку зчитує температуру в градусах Цельсія. Далі вона має вивести еквіваленту температуру в градусах Кельвіна та Фаренгейта.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  c = int(input("Температура(в цельсія): "))  f = round((9/5)\*c)+32  k = c + 273  print("Цельсій", "%18s" % "Фаренгейт", "%18s" % "Кельвін")  print("%4s"%c,"%9s"% " - ", "%7s"% f, "%11s"% " - ", "%8s"%k)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скрнішот: | 0,5 бала |
| 6. | Умова:  Напишіть програму, яка зчитуватиме 4-значне ціле число та буде виводити суму його цифр. Наприклад, якщо ввести 3141, програма має вивести 9 (=3+1+4+1).  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  while True:  number = input("Введіть 4-значне число: ")  if len(number) == 4:  break  suma = sum([int(i) for i in number])  print(suma)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 7. | Умова:  Поширеною думкою є те, що 1 рік людського життя еквівалентний 7 рокам життя собаки. Проте це просте правило перестає виконуватись, коли собака стає дорослою (приблизно в 2 роки). У результаті деякі люди вважають, що краще вважати перші 2 роки життя собаки еквівалентними 10.5 рокам життя людини, а наступні – 4м рокам людини.  Напишіть програму, яка реалізує переведення з людських років у собачі за даними правилами. Ваш додаток має виводити попередження, коли вводиться від’ємне число.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  while True:  years = float(input("Людських років: "))  if years >= 0:  if years <= 1:  print("Собачих років:",years\*7)  elif 1 < years <= 2:  print("Собачих років:", 7 + 1.75\*years)  else:  print("Собачих років:", 10.5 + (years-2)\*4)  break  else:  print("Число не може бути від'ємним")    printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1,5 бали |
| 8. | Умова:  Онлайн-ритейлер продає 2 товари: штучки та штукенції. Кожна штучка важить 75г. Кожна штукенція важить 112г. Напишіть програму, яка зчитує кількість штучок та кількість штукенцій в замовленні від користувача. Потім програма має обчислювати та виводити на екран загальну масу замовлення.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  shtuka = int(input("Кількість штукчок: "))  shtukencia = int(input("Кількість штукенцій: "))  veight = shtuka\*75 + shtukencia\*112  print(veight/1000,"кг")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 9. | Умова:  Багато людей думають про свій зріст у футах і дюймах, але у нас використовується метрична система. Напишіть програму, яка зчитує зріст у сантиметрах, а виводить у футах і дюймах.  Підказка: 1 фут = 12 дюймів. 1 дюйм = 2.54 см.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  cm = float(input("Введіть зріст у см: "))  d = cm / 2.54  f = d / 12  print("Дюйми - %.3f" % d,'\n'+"Фути - %.3f" % f)  f = int(d // 12)  d = round(d-f\*12)  print("Ваш зріст {} футів і {} дюймів".format(f,d))  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 10. | Умова:  Напишіть програму, яка зчитує проміжок часу у вигляді кількості днів, годин, хвилин та секунд. Обчисліть та виведіть загальну кількість секунд у цьому проміжку часу.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  print("Проміжо часу: ")  days1 = int(input("Дні: "))  hours1 = int(input("Години: "))  minutes1 = int(input("Хвилини: "))  seconds1 = int(input("Секунди: "))  sec = datetime.timedelta(days = days1, hours = hours1, minutes = minutes1, seconds = seconds1)  print(sec.total\_seconds())  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1 бал |
| 11. | Умова:  Напишіть програму, яка зчитує 3 цілих числа і сортує їх у порядку від меншого до більшого. Використовуйте функції min() та max() для знаходження найменшого та найбільшого значень. Значення між ними можна обчислити, знайшовши суму всіх 3 чисел та віднявши мінімум та максимум.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  numbers=[(int(input("Введіть число: "))) for x in range(3)]  print("Максимальне значення: ", max(numbers))  print("Середнє значення: ", sum(numbers)-min(numbers)-max(numbers))  print("Мінімальне значення : ", min(numbers))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1 бал |
| 12. | Умова:  Створіть програму, яка обчислює середнє для набору значень, введеного користувачем. Користувач буде вводити 0 як індикатор того, що більше значень вводитись не буде. Ваша програма має виводити відповідне повідомлення про помилку, якщо перше введене значення буде нулем. Ця вправа не передбачає використання непримітивних типів даних.  Підказка: оскільки 0 позначає кінець вводу, він має не включатись в обчислення середнього.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  numbers=[]  x = int(input("Введіть значення: "))  try:  while x != 0:  numbers.append(x)  x = int(input("Введіть значення: "))  print("Середнє значення: ", sum(numbers) / len(numbers))  except ZeroDivisionError:  print("Помилка!")  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1 бал |
| 13. | Умова:  Уявімо, що Ви тільки-но відкрили новий накопичувальний рахунок зі ставкою 14% річних. Отриманий прибуток виплачується в кінці року та додається до балансу рахунку. Напишіть програму, що починає роботу зі зчитування кількості покладених на рахунок грошей. Далі вона має обчислити та вивести на екран суму на рахунку через 1, 2 і 3 роки. Показ сум обмежується 2 знаками після крапки.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  money = float(input("Введіть гроші, які покладите на рахунок: "))  for year in range(1,4):  money = money \* 1.14  print("Грошей на балансі через {} рік: {:.2f}".format(year, money))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1 бал |
| 14. | Умова:  Пекарня продає буханки хлібу за 8.50 грн. На вчорашній хліб встановлено знижку на 60%. Напишіть програму, яка починає зі зчитування кількості буханок вчорашнього хлібу, які хоче придбати користувач. Потім програма має вивести на екран звичайну вартість товару, скидку за те, що він вчорашній, а також загальну суму покупки. Всі значення мають виводитись двома десятковими знаками після крапки та вирівнюватись по правому краю.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  bukhanka = int(input("Ведіть кількість буханок: ".rjust(46)))  print("Звичайна вартість буханок: {:.2f}".rjust(53).format(bukhanka\*8.50))  print("Вартість буханок зі скидкою: {:.2f}".rjust(55).format(bukhanka\*8.50\*0.6))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1 бал |
| 15. | Умова:  Напишіть програму, яка зчитує ціле число n від користувача та показує суму всіх цілих чисел від 1 до n. Формула для знаходження цієї суми:  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  n= int(input("Введіть число: "))  print("Сумма чисел: ", (n\*(n+1))/2)  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 0,5 бала |
| 16. | Умова:  Напишіть програму, яка починає зі зчитування радіусу r від користувача. Програма продовжуватиме роботу, обислюючи та виводячи на екран площу кругу з радіусом r та об’єм кулі з радіусом r. Використовуйте константу pi з модуля math.  Код:  import math, datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  r = float(input("Введіть радиус: "))  print("Площа круга: ", math.pi\*r\*\*2)  print("Об'єм кулі: ", math.pi\*r\*\*3)  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 0,5 бала |
| 17. | Умова:  Полігон називають регулярним, якщо всі його сторони мають однакові довжини, а кути між усіма суміжними сторонами теж однакові. Площу регулярного полігону можна обчислити за формулою  де s – довжина сторони, а n – кількість сторін. Напишіть програму, яка зчитує ці значення і виводить на екран площу сконструйованого за даними значеннями полігону.  Код:  import math,datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  n= int(input("Введіть кількість сторін: "))  s= int(input("Введіть довжину сторони: "))  if n<3 or s<=0:  print("Некорктні дані!")  else:  print("Площа цьго полігону: ", (n\*math.pow(s,2))/(4\*math.tan(math.pi/n)))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 0,5 бала |
| 18. | Умова:  Напишіть програму, яка починає роботу зі зчитування кількості секунд від користувача. Далі вона має вивести еквівалентний проміжок часу у формі Д:ГГ:ХХ:СС (дні, години, хвилини, секунди). Потрібно, щоб форматований вивід передбачав по 2 знакових місця для годин, хвилин та секунд з дописуванням незначущого нуля за потреби.  Код:  import math, datetime, time  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  sec= float(input("Введіть секунди: "))  vremya= time.gmtime(sec)  print(time.strftime('{}:{}:{}:{}'.format(vremya[2]-1,vremya[3] if vremya[3]> 9 else "0"+str(vremya[3]),vremya[4] if vremya[4]> 9 else "0"+str(vremya[4]),  vremya[5] if vremya[5]> 9 else "0"+str(vremya[5]))))  printTimeStamp("Даніїл Коробов та Дудник Віталій" )  Скріншот: | 1,5 бали |
| 19. | Умова:  Припустимо, розумна програма може не вмикати будильник, щоб людина виспалась на вихідних чи у відпустці. Розробіть додаток, який буде приймати на вхід від користувача позначки про вихідний та відпустку, а виводитиме рішення, чи вмикати будильник. Вмикати його не слід, якщо день вихідний або людина у відпустці.  Код:  import math,datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  week= {"понеділок": True,"вівторок": True,"середа":True,"четвер":True,"п'ятниця":True,"субота":False,"неділя":False}  try:  holiday = input("Введіть день відпустки(якщо її немає то '0'): ").lower()  week[holiday] = False  day = input("Введіть день тиждня: ").lower()  if week[day]:  print("Дзінь-дзілінь!")  else:  print("Насолоджуйся сном))")  except KeyError:  print("Такого дня не існує!")  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1 бал |
| 20. | Умова:  Напишіть Python-скрипт, який після запуску виведе:    Використовуйте циклічні оператори для отримання такого виводу  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  propuski=[7,4,2,1,0]  c = 3  for x in range(5):  print(propuski[x]\*" "+"x"\*c)  c+= (propuski[x]- propuski[x+(1 if x<4 else 0)])\*2  for x in range(1,6):  print(propuski[-x]\*" "+"x"\*c)  c-= ((propuski[-x-1] if x<5 else 10) - propuski[-x])\*2  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1,5 бала |
| 21. | **Умова:**  **Готуємо ідеальне яйце.** Під час варіння протеїни яйця спочатку денатурують (змінюють природну структуру), а потім коагулюють (згортаються). Коли температура досягає критичної точки, хімічні реакції розпочинаються і протікають швидше з ростом температури. Білок яйця коагулює при температурі близько 63оC, а жовток – за температури понад 70оC. Для варки рідкого яйця потрібно достатньо довго нагрівати його при температурі від 63 оC до 70оC. Для приготування крутого яйця дозволено досягати температури понад 70 оC.  Наступна формула виражає час (в секундах), коли центр жовтка досягне температури (у градусах Цельсія):  де M – маса яйця, – густина, – питома теплоємність, а K – теплопровідність яйця. Релевантними значеннями для малого яйця є маса M = 47г та M = 67г для великого яйця, 1.038 , , and . За умовами задачі – температура киплячої води (у градусах Цельсія), а – початкова температура (у градусах Цельсія) яйця перед його покладанням у воду. Реалізуйте формулу в програмі, задавши , та обчисліть для великого яйця, взятого з холодильника () та при кімнатній температурі ().  Код:  import math,datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  t0= 4  print("При температурі 4 градуси",( 67\*\*(2/3)\*3.7\*1.038\*\*(1/3) ) / 5.4\*math.pi\*\*2\*((4\*math.pi/3)\*\*(2/3)) \* math.log(0.76\*((t0-100)/-30)))  t0= 20  print("При температурі 20 градусів",( 67\*\*(2/3)\*3.7\*1.038\*\*(1/3) ) / 5.4\*math.pi\*\*2\*((4\*math.pi/3)\*\*(2/3)) \* math.log(0.76\*((t0-100)/-30)))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1,5 бала |

**Рівень В**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Умова:  Закон ідеального газу – математичне наближення поведінки газів при зміні тиску, об’єму та температури. Зазвичай він описується рівнянням  де P – тиск у паскалях (Па), V – об’єм у літрах, n – молярна маса речовини, R = 8.314 mol K J – константа ідеального газу, T – температура в градусах Кельвіна.  Напишіть програму, яка обчислює молярну масу газу, коли користувач задає тиск, об’єм і температуру. Протестуйте свою програму на визначення кількості молей газу в SCUBA tank. Типовий SCUBA tank містить 12 літрів газу під тиском 20,000,000 Па. Беремо кімнатну температуру (20º).  Підказка: Температура конвертується з Цельсія в Кельвіни шляхом додавання 273.15 градусів.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  p= float(input("Введіть тиск(Па): "))  v= float(input("Введіть об'єм(л): "))/1000  t= float(input("Введіть температуру(С): "))+273.15  if p<=0 or v<=0 or t<=0:  print("Некоректні дані!")  else:  print("Молярна маса речовини(г/моль): ", (p\*v)/ (8.314 /t))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 2 бали |
| 2. | Умова:  Нижче наведена таблиця містить діапазони магнітуд землетрусів за шкалою Ріхтера та опис відповідного землетрусу:   |  |  | | --- | --- | | Магнітуда | Опис | | <2.0 | Мікро (micro) | | 2.0-3.0 | Дуже слабкий (very minor) | | 3.0-4.0 | Слабкий (minor) | | 4.0-5.0 | Легкий (light) | | 5.0-6.0 | Помірний (moderate) | | 6.0-7.0 | Сильний (strong) | | 7.0-8.0 | Дуже сильний (major) | | 8.0-10.0 | Великий (great) | | >=10.0 | Рідкісно великий (meteoric) |   Напишіть програму, яка зчитує від користувача магнітуду та виводить відповідний опис. Докладніше [тут](https://uk.wikipedia.org/wiki/%D0%A8%D0%BA%D0%B0%D0%BB%D0%B0_%D0%A0%D1%96%D1%85%D1%82%D0%B5%D1%80%D0%B0).  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  mag= float(input("Введіть магнітуду: "))  if 0<mag<2:  print("Мікро: Мікроземлетруси, не відчуваються.")  if 2<=mag<3:  print("Дуже слабкий: Як правило не відчуваються, але реєструються.")  if 3<=mag<4:  print("Слабкий: Часто відчуваються, дуже рідко завдають шкоди.")  if 4<=mag<5:  print("Легкий:Відчутне тремтіння речей всередині будинків, значна шкода малоймовірна.")  if 5<=mag<6:  print("Помірний : Може завдати значної шкоди старим та погано сконструйованим будівлям на незначній території. Щонайбільше, незначні пошкодження добре спроектованим будівлям.")  if 6<=mag<7:  print("Сильний : Може спричинити руйнацію на території до 150 км довжиною/шириною в населених регіонах.")  if 7<=mag<8:  print("Дуже сильний : Значна руйнація на значній території.")  if 8<=mag<9:  print("Великий:Серйозна руйнація на територіях довжиною/шириною в сотні кілометрів.")  if mag>9:  print("Рідкісно великий!!!")  else:  print("Неверні дані!")  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 1,5 бали |
| 3. | Умова:  Деякий телефонний тариф включає 200 хвилин в мережі та 50 текстових повідомлень за ₴35.00 на місяць. Кожна додаткова хвилина коштує ₴0.17, а додаткова SMS - ₴0.15. Всі рахунки включають ₴0.44 внеску в Пенсійний фонд та 5%-вий податок.  Напишіть програму, яка зчитує кількість хвилин та текстових повідомлень, використаних за місяць. Виведіть базову плату за користування (без внесків та податків), а також загальний рахунок для користувача. Забезпечте виведення грошових сум з 2 знаками після коми.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  min= int(input("Введіть кіл-сть витрачених хвилин: "))  mes= int(input("Введіть кіл-сть написаних повідомлень: "))  rakhunok=35  if min> 200:  rakhunok+= (min - 200) \* 0.17  if mes > 50:  rakhunok+= (mes - 50) \* 0.15  print("Рахунок з податками: {:.2f}".format(rakhunok))  print("Чистий рахунок: {:.2f}".format(rakhunok-0.44-(rakhunok\*0.05)))  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 2 бали |
| 4. | Умова:  13-цифровий міжнародний стандартний номер книги (International Standard Book Number) – це унікальний код, який ідентифікує книгу, що продається. Остання цифра називається перевірочною цифрою (check digit) і використовується для перевірки на помилки. Для обчислення цього значення   * 1. кожну з 12 перших цифр почережно множать на 1 або 3,   2. додають добутки та ділять результат за модулем 10. Перевірочна цифра отримується відніманням від 10 отриманого результату.   Наприклад, для ISBN 9789609319614    Напишіть програму, що зчитуватиме 13-цифровий ISBN та перевірятиме останню цифру на валідність цього номеру.  Код:  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  num=[]  isbn= input("Введіть номер ISBN: ")  while len(isbn) != 13:  print("У номері повинно бути 13 цифр!")  isbn = int(input("Введіть номер ISBN: "))  for i in range(12):  if i%2==0:  num.append(int(isbn[i]))  else:  num.append(int(isbn[i])\*3)  if 10-(sum(num)%10)== int(isbn[12]):  print("Номер валідний!")  else:  print("Номер не валідний!")  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 2,5 бала |
| 5. | Умова:  Розгляньте програмне забезпечення, яке працює на апараті для приготування кави. Одна із задач, яку воно повинно виконувати – визначення того, скільки здачі повертати з готівкової оплати товару.  Напишіть програму, яка починає роботу зі зчитування кількості копійок (ціле число). Далі програма має обчислювати та виводити на екран номінали монет та їх кількість для того, щоб видати здачу покупцю. Здача має віддаватись якомога меншою кількістю монет. Припускаємо, що машина завантажена монетами номіналами 25, 50 копійок, 1, 2 та 5 грн.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  while True:  coins = int(input("Кількість копійок: "))  if coins > 0:  break  fifty\_cent = coins // 50  twenty\_five\_\_cent = (coins % 50) // 25  five\_cent = ((coins % 50) % 25 )// 5  two\_cent = (((coins % 25) % 10) % 5) // 2  one\_cent = (((coins % 25) % 10) % 5) % 2  print("50:", fifty\_cent, "\n" +  "25:", twenty\_five\_\_cent, "\n" +  "5:", five\_cent, "\n" +  "2:", two\_cent, "\n" +  "1:", one\_cent)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2,5 бала |
| 6. | Умова:  Напишіть програму, яка рекомендуватиме людині домашнього улюбленця відповідно до стилю життя його майбутнього власника.   |  |  |  | | --- | --- | --- | | **Місце проживання** | **Час вдома** | **Рекомендація** | | Будинок | Понад 18 годин | В’єтнамське порося | | Будинок | Від 10 до 17 годин | Собака | | Будинок | Менше 10 годин | Змія | | Квартира | Більше 10 годин | Кішка | | Квартира | Менше 10 годин | Хом’як | | Гуртожиток | Більше 6 годин | Рибки | | Гуртожиток | Менше 6 годин | Мурашник |   Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  time = int(input("Час вдома: "))  home = input("Місце проживання: ")  if home == "Будинок" and time >= 18:  print("Вам підходить в'єтнамське порося")  elif home == "Будинок" and 10<= time <= 17 :  print("Вам підходить собака")  elif home == "Будинок" and time < 10 :  print("Вам підходить змія")  elif home == "Квартира" and time >= 10:  print("Вам підходить кішка")  elif home == "Квартира" and time <10:  print("Вам підходить хом'як")  elif home == "Гуртожиток" and time >= 6 :  print("Вам підходять рибки")  elif home == "Гуртожиток" and time < 6 :  print("Вам підходить мурашник")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2,5 бала |
| 8. | Умова:  У деяких штатах США старіші номерні знаки складаються з 3 великих літер, за якими йдуть 3 цифри. Коли всі можливі номерні знаки були вичерпані, перейшли до формату «4 цифри + 3 великі букви». Напишіть програму, яка зчитує рядок символів, а потім визначає, чи підходить введений номер під якийсь із форматів. Якщо це не так, слід повідомити користувача про помилку.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  import re  format1 = "[A-Z]{3}[0-9]{3}"  format2 = "[0-9]{4}[A-Z]{3}"  row\_sumbol = input("Введіть рядок для перевірки:")  match1 = re.match(format1, row\_sumbol)  match2 = re.match(format2, row\_sumbol)  if match1:  print("Співпало з форматом: 3 великі літери, 3 цифри")  elif match2:  print("Співпало з форматом: 4 цифри, 3 великі літери")  else:  print("Не співпадає з шаблоном")  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 2,5 бала |
| 9. | Умова:  Напишіть програму, яка конвертує десяткове число в двійкове. Зчитуйте десяткове число від користувача як ціле число, а потім використовуйте алгоритм ділення для виконання перетворення.  **Ініціалізувати змінну result порожнім рядком**  **Оголосити змінну q – число для перетворення**  **repeat**  **Оголосити r та присвоїти йому значення остачі від ділення q на 2**  **Звести r до рядкового типу та додати її до початку**  **result**  **Поділити націло q на 2 та зберегти результат у q**  **until q не доівнює 0**  Коли алгоритм завершує роботу, змінна result міститиме двійкове представлення числа. Відобразіть результат на екрані з відповідним пояснювальним текстом.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  result = ""  q = int(input("Введіть число для конвертації: "))  while q != 0:  r = q % 2  result += str(r)  q //=2  print("Двійкове представлення:",result)  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 2,5 бали |
| 10. | Умова:  Поверхня Землі викривлена, і відстань між градусами довготи варіюється залежно від широти. У результаті, знаходження відстані між 2 точками на поверхні Землі дещо ускладнюється.  Нехай (t1, g1) та (t2, g2) – широти і довготи 2 точок на поверхні планети. Відстань між ними в кілометрах визначається за формулою:  **Відстань = 6371.01 × arccos(sin(t1) × sin(t2) + cos(t1) × cos(t2) × cos(g1 − g2))**  Величина 6371.01 – середній радіус планети Земля в кілометрах.  Напишіть додаток, що дозволятиме користувачу вводити широту і довготу 2 точок у градусах. Ваша програма має виводити відстань між точками в кілометрах.  Підказка: тригонометричні функції в Python працюють з радіанною мірою. Як результат, буде потрібно конвертувати користувацький ввід з градусів у радіани до обчислення відстані. Модуль math містить функцію radians(), яка переводить градуси в радіани.  Код:  import math,datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  t1 = math.radians(float(input("Широта т.1(в градусах): ")))  g1 = math.radians(float(input("Довгота т.1(в градусах): ")))  t2 = math.radians(float(input("Широта т.2(в градусах): ")))  g2 = math.radians(float(input("Довгота т.2(в градусах): ")))  distance = 6371.01 \* math.acos(math.sin(t1) \* math.sin(t2) + math.cos(t1) \* math.cos(t2) \* math.cos(g1 - g2))  print("Відстань",round(distance,2),"км")  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 3 бали |
| 11. | Умова:  На основі попередньої задачі 6 визначте площу трикутника, утвореного містами Львів, Одеса та Харків. Пропонується визначити відстань між цими містами, а потім встановити площу за формулою Герона.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  import math  def dist(t1,g1,t2,g2):  distance = 6371.01 \* math.acos(math.sin(t1) \* math.sin(t2) + math.cos(t1) \* math.cos(t2) \* math.cos(g1 - g2))  return distance  lviv\_t = math.radians(49.83826)  lviv\_g = math.radians(24.02324)  harkiv\_t = math.radians(49.98081)  harkiv\_g = math.radians(36.25272)  odesa\_t = math.radians(46.47747)  odesa\_g = math.radians(30.73262)  dist1 = dist(lviv\_t, lviv\_g, harkiv\_t, harkiv\_g)  dist2 = dist(harkiv\_t, harkiv\_g, odesa\_t, odesa\_g)  dist3 = dist(lviv\_t, lviv\_g, odesa\_t, odesa\_g)  p = (dist1 + dist2 + dist3)/2  area = math.sqrt(p\*(p-dist1)\*(p-dist2)\*(p-dist3))  print("Площа трикутника між Одесою, Харківом та Львівом:",round(area, 2),"км")  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 2 бали  В10 |
| 12. | Умова:  Китайський гороскоп ставить у відповідність 12-річному циклу назви тварин:    Напишіть програму, яка зчитує рік і виводить тварину, що йому відповідає. Програма має коректно працювати з роками нашої ери, а не лише з представленими в таблиці.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  year = int(input("Введіть рік: "))  animals = {  2000:"Dragon", 2001:"Snake", 2002:"Horse", 2003:"Sheep", 2004:"Monkey", 2005:"Rooster",  2006:"Dog", 2007:"Pig", 2008:"Rat", 2009:"Ox", 2010:"Tiger", 2011:"Hare"}  if year in animals:  print(animals.get(year))  else:  if year < 2000:  while year not in animals:  year += 12  print(animals.get(year))  else:  while year not in animals:  year -= 12  print(animals.get(year))  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2,5 бали |
| 13. | Умова:  Аутентичність пристрою можна перевірити за допомогою отриманої з IMEI цифри Луна, порівнявши її з цифрою виробниками. Отримати цифру Луна можна таким чином:   * + Додати 14 цифр ІМЕІ так, що цифри на парних позиціях подвоюються при входженні в суму (якщо утворюється дворозрядне число, треба додати його розряди), а на непарних – залишаються з тим же значенням. Наприклад, для ІМЕІ 357683036257378   + Якщо останній розряд суми дорівнює 0, це і є цифра Луна. Інакше потрібно відняти отримане значення від 10.     Звідси, цифра Луна – 10-2=8. Перевірте введений з клавіатури ІМЕІ на відповідність виробнику з цифрою Луна 6.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  while True:  imei = input("IMEI: ")  if len(imei) == 15:  break  else:  print("Ви ввели не 15 символів, а", len(imei))  odd = [int(i) for i in imei[:14:2]]  even = [int(i)\*2 for i in imei[1:14:2]]  even = [int(str(i)[0])+int(str(i)[1]) if i > 9 else i for i in even]  suma = str(sum(even)+ sum(odd))  if int(suma[-1]) == 0:  print("Цифра Луна - 0")  else:  lun = 10 - int(suma[-1])  print("Цифра Луна -", lun)  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бала |
| 14. | Умова:  Коли дмухає вітер в холодну погоду, повітря відчувається ще холоднішим, ніж є насправді. Це відбувається тому, що рух повітря пришвидшує охолодження теплих об’єктів, таких як люди. Даний ефект називають прохолодністю вітру.  У 2001 році в Канаді, Великобританії та США було впроваджено формулу, яка обчислює індекс прохолодності вітру (wind chill index):  У даній формулі – температура повітря в градусах Цельсія, – швидкість вітру в км/год.  Напишіть програму, яка спочатку зчитує температуру повітря та швидкість вітру, а потім виводить індекс прохолодності вітру, округлений до найближчого цілого.  Індекс прохолодності вітру вважається коректним за умови, якщо температура не перевищує 10ºС, а швидкість вітру буде понад 4.8 км/год. Дані умови потрібно врахувати при виводі інформації.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  v = float(input("Швидкість вітру(км/год): "))  t = int(input("Температура вітру(в градусах Цельсія): "))  if t >10 or v > 4.8:  if v > 4.8:  print("Індекс некоректний для швидкості вітру більше 4.8 км/год")  if t >10:  print("Індекс некоректний для температури більше 10 градусів Цельсія")  else:  WCI = 13.12 + 0.6215 \* t - 11.37 \* v + 0.3965 \* t \* v  print("Індекс прохолодності вітру",round(WCI))  printTimeStamp("Віталій Дудник Коробов Даніїл")  Скріншот: | 1,5 бали |
| 15 | Умова:  Високосні роки визначаються за наступними правилами:  • Будь-який рік, що націло ділиться на 400, високосний.  • З решти всі роки, що діляться на 100 – невисокосні.  • З решти всі роки, що діляться на 4, є високосними.  • Всі інші - невисокосні.  Напишіть програму, яка зчитує рік та виводить повідомлення про те, чи є цей рік високосним.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  year = int(input("Рік: "))  if year % 400 == 0:  print("Високосний")  elif year % 100:  print("Невисокосний")  elif year % 4:  print("Високосний")  else:  print("Невисокосний")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 1,5 бали |
| 16. | Умова:  Напишіть програму, яка обчислює периметр багатокутника. Починайте з вводу координат (x; y) першої точки і продовжуйте ввід для інших точок. Закінчення вводу відбудеться при введенні порожнього рядка в х-координату. Кожного разу при зчитуванні координат нової точки слід обчислювати відстань від попередньої точки до неї та додавати результат до периметру. Коли введено порожній рядок, програма має обчислити відстань від першої точки до останньої та додати її до периметру. Після цього слід вивести повний периметр. Приклад такого вводу показано нижче:  Enter the x part of the coordinate: 0  Enter the y part of the coordinate: 0  Enter the x part of the coordinate: (blank to quit): 1  Enter the y part of the coordinate: 0  Enter the x part of the coordinate: (blank to quit): 0  Enter the y part of the coordinate: 1  Enter the x part of the coordinate: (blank to quit):  The perimeter of that polygon is 3.414213562373095  Код:  import math  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  koordinaty=[]  dlina=[]  for a in range(3):  x= int(input("Enter the x part of the coordinate:"))  y= int(input("Enter the y part of the coordinate:"))  koordinaty.append((x,y))  try:  while x!= " ":  x = input("Enter the x part of the coordinate(blank to quit): ")  y = input("Enter the y part of the coordinate(blank to quit): ")  koordinaty.append((int(x),int(y)))  except ValueError:  for i in koordinaty:  try:  dlina.append(math.sqrt((koordinaty[koordinaty.index(i)+1][0]-i[0])\*\*2+(koordinaty[koordinaty.index(i)+1][1]-i[1])\*\*2))  except IndexError:  dlina.append(math.sqrt((koordinaty[0][0] - i[0]) \*\* 2 + (koordinaty[0][1] - i[1]) \*\* 2))  print("Периметер:",sum(dlina))  printTimeStamp("Коробов Даніїл та Віталій Дудник")  Скріншот: | 2,5 бали |
| 17. | Умова:  Кількість енергії, необхідної для збільшення температури 1 граму матеріалу на 1ºС, називається теплоємністю C. Загальна енергія, потрібна для підняття температури m грамів матеріалу на ΔT градусів Цельсія можна обчислити за формулою:  Напишіть програму, яка зчитує масу води та температурну зміну від користувача. Ваша програма має відображати загальну кількість енергії, яку потрібно додати чи забрати, щоб досягнути заданої зміни температури.  Підказка: питома теплоємність води складає 4.186 Дж/г ºС. Оскільки вода має густину 1.0 г/мл, можна використовувати як грами, так і мілілітри в даному випадку.  Розширьте свою програму, щоб вона також обчислювала вартість нагріву води. Оплата електроенергії зазвичай відбувається в кВт год, а не в джоулях. У даній вправі слід припустити, що електроенергія коштує 1.33 грн за кВт-год. Використовуйте програму, щоб обчислити вартість кип’ятіння 1 чашки води для кави.  Підказка: слід розглянути множник, що дозволяє конвертувати джоулі в кВт-год, щоб завершити виконання вправи.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  m = float(input("Маса води(г/мл): "))  T = float(input("Зміна температури:"))  C = 4.186  q = m\*T\*C  print("Кількість необхідної енергії:", q)  elect = 1.33  kBt\_hour = q / (3.6 \* 10\*\*6)  price = elect \* kBt\_hour  print("%.2f" % price)  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 2,5 бали |
| 18. | Умова:  Позиції на шаховій дошці визначаються буквою (колонка) та цифрою (рядок), як показано на рисунку.    Напишіть програму, яка зчитує позицію від користувача. Використовуйте оператор if, щоб визначити, чи колонка починається з білої або чорної клітини. Далі використайте модулярну арифметику, щоб повідомити колір клітинки в цьому рядку. Наприклад, при вводі a1 програма має повідомити, що клітинка чорна. Якщо користувач вводить d5, програма виводить повідомлення про те, що комірка біла. Ваш додаток повинен перевіряти, чи введені дані коректні (не виходять за межі дошки), і якщо ні – виводити на екран повідомлення про помилку.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  bukva= {"a":1,"b":2,"c":3,"d":4,"e":5,"f":6,"g":7,"h":8}  poza= input("Введіть позіцію на дощці: ")  try:  if 0>=int(poza[0]) or int(poza[0])>=9:  print("Некоректний ввід!")  elif (int(poza[0])%2== 1 and bukva[poza[1]]%2==1) or (int(poza[0])%2== 0 and bukva[poza[1]]%2==0):  print("Квадрат чорний!")  else:  print("Квадрат білий!")  except KeyError:  print("Некоректний ввід!")  printTimeStamp("Коробов Даніїл та Віталій Дудник")  Скріншот: | 2,5 бали |
| 19. | Умова:  У деякому університеті використовується буквенна система оцінювання у відповідності до балів із таблиці:   |  |  | | --- | --- | | **Буквенна оцінка** | **Бальна оцінка** | | A+ | >4.0 | | A | 4.0 | | A- | 3.7 | | B+ | 3.3 | | B | 3.0 | | B- | 2.7 | | C+ | 2.3 | | C | 2.0 | | C- | 1.7 | | D+ | 1.3 | | D | 1.0 | | F | 0 |   Напишіть програму, яка зчитуватиме буквенну оцінку від користувача. Далі вона виводить на екран бальну оцінку, що відповідає введеній буквенній. Забезпечте коректну роботу для ситуації, коли введена буквенна оцінка не передбачена в таблиці.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  mark = input("Введіть буквену оцінку: ")  if mark == "A+":  print(5.0)  elif mark == "A":  print(4.0)  elif mark == "A-":  print(3.7)  elif mark == "B+":  print(3.3)  elif mark == "B":  print(3.0)  elif mark == "B-":  print(2.7)  elif mark == "C+":  print(2.3)  elif mark == "C":  print(2.0)  elif mark == "C-":  print(1.7)  elif mark == "D+":  print(1.3)  elif mark == "D":  print(1.0)  elif mark == "F":  print(0)  else:  print("Немає такої оцінки")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 20. | Умова:  Попередня вправа включала таблицю, яка показувала правила конвертації буквенної оцінки в бали для деякої академічної установи. У даній вправі Вам потрібно написати програму, яка дозволятиме користувачеві вводити оцінки студентів групи, доки не буде введено -1. У процесі програма має конвертувати оцінку в її буквенний еквівалент на виводити на екран. Наприкінці програма має обчислити середню оцінку по групі, вивести її та її буквенний еквівалент.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  def ABCDF(mark):  if 4 < mark <= 5:  return "A+"  elif 3.7 < mark <=4:  return "A"  elif 3.3 < mark <= 3.7 :  return "A-"  elif 3.0 < mark <= 3.3:  return "B+"  elif 2.7< mark <= 3.0:  return "B"  elif 2.3< mark < 2.7:  return "B-"  elif 2.0 < mark <= 2.3:  return "C+"  elif 1.7 < mark <2.0:  return "C"  elif 1.3 < mark <= 1.7:  return "C-"  elif 1.0 < mark <= 1.3:  return "D+"  elif 0 < mark <= 1.0:  return "D"  elif mark == 0:  return "F"  else:  return"Немає такої оцінки"  marks = []  while True:  mark = int(input("Оцінка: "))  if mark == -1:  break  marks.append(mark)  print("Буквений варіант:",ABCDF(mark))  average = sum(marks)/len(marks)  print("Середній бал по групі:",str(average)+". Буквений варіант:",ABCDF(average))  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали  В19 |
| 21. | Умова:  Найбільший спілдьний дільник двох додатніх цілих чисел (n, m) є найбільшим числом d, на яке вони діляться без остачі. Один з алгоритмів для знаходження НСД:  **Ініціалізувати d значенням меншого з чисел m і n.**  **While на d націло не діляться m і n do**  **Зменшити значення d на 1**  **Вивести d**  Напишіть програму, яка зчитує 2 цілих додатніх числа та знаходить найбільший спільний дільник за алгоритмом.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  def func(n,m):  d = min(n,m)  while n % d != 0 or m % d != 0:  d -= 1  return d  n = int(input("Введіть 1-ше число: "))  m = int(input("Введіть 2-ше число: "))  print("НСД:",func(n,m))  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 22. | Умова:  Електромагнітне випромінювання можна розділити на 7 категорій за частотою:    Напишіть програму, яка зчитує частоту радіації та виводить відповідну назву хвиль.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  frequency = int(input("Enter frequency:"))  if 0 < frequency < 3 \* 10\*\*9:  print("Radio waves")  elif 3 \* 10\*\*9 < frequency < 3 \* 10\*\*12:  print("Microwaves")  elif 3 \* 10\*\*12< frequency < 4.3 \* 10\*\*14:  print("Infrared light")  elif 4.3 \* 10\*\*14 < frequency < 7.5 \* 10\*\*14:  print("Visiable light")  elif 7.5 \* 10\*\*14 < frequency < 3 \* 10 \*\*17:  print("Ultraviolet light")  elif 3 \* 10\*\*17 < frequency < 3 \*10\*\*19:  print("X-rays")  else:  print("Gamma rays")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 23. | Умова:  Компанія, що займається постачанням води, обчислює клієнтські рахунки за наступною формулою:   1. Фіксований внесок – 20 грн. 2. За перші 30 кубометрів плата складає 9.86 грн./м3 3. За наступні 20 кубометрів – 11.22 грн./м3 4. За наступні 10 кубометрів – 13.06 грн./м3 5. За додаткові кубометри – 17.89 грн./м3   Напишіть програму, яка буде зчитувати споживання води в кубометрах та виводити на екран рахунок.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  n = int(input("Спожито води: "))  first = lambda n: n \* 9.86 / 30  second = lambda n: n \* 11.22 / 20  third = lambda n: n \* 13.06 / 10  fourth = lambda n: n \* 17.89  if n <= 30:  print("%.2f" % first(n), "грн") + 20  elif 30 < n <= 50:  price = second(n - 30) + first(30) + 20  print("%.2f" % price, "грн")  elif 50 < n <= 60:  price = third(n - 50) + second(20) + first(30)+ 20  print("%.2f" % price, "грн")  else:  price = fourth(n-60) + third(10) + second(20) + first(30) + 20  print("%.2f" % price, "грн")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2,5 бали |
| 24. | Умова:  Наступна таблиця перелічує рівень гучності в децибелах для кількох поширених шумів. Напишіть програму, яка зчитує рівень шуму в децибелах від користувача. Якщо користувач вводить рівень шуму, що співпадає з табличним значенням, виведіть повідомлення, що містить лише тип шуму. Якщо користувач вводить кількість децибелів, яка знаходиться в проміжку між шумами, програма повідомлятиме, між якими шумами розташовується даний рівень шуму. Забезпечте коректну роботу програми при значеннях, нижчих за найтихіший шум, та значеннях, що перевищують найгучніший шум з таблиці.    Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  n = int(input("Enter number: "))  noise = {130: "Jackhammer", 106: "Gasvlawnmover", 70: "Alarm clock", 40: "Quiet room"}  if n in noise:  print("Noise type: ",noise.get(n))  elif n > 130 :  print("Шум вищий за табличний")  elif n < 40:  print("Шум нижчий за табличний")  else:  maks = n  min = n  while maks not in noise:  maks += 1  while min not in noise:  min -= 1  print("Noise type between",noise.get(maks),"and",noise.get(min))  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 25. | Умова:  Напишіть програму, яка відобразить табличку множення 10х10, як показано на рисунку:    Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  for row in range(11):  if row == 0:  print(" ",end = "")  for column in range(1, 11):  print(str(column).rjust(4), end="")  else:  print(str(row).rjust(2), end = "")  for column in range(1,11):  print(str(column\*row).rjust(4), end="")  print("")  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 26. | Умова:  Факторизація цілого числа n може виконуватись за алгоритмом:  **factor = 2**  **While factor <= n do**  **If n націло ділиться на factor then**  **значить factor є множником n**  **поділити n на factor націло,**  **Else**  **збільшити factor на 1**  Напишіть програму, яка зчитує ціле число . Якщо число менше 2, програма має повідомити про помилку. Інакше додаток обчислює прості числа, на які можна розкласти . Наприклад:    Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  factor = 2  try:  n = int(input("Enter the integer(2 or greater): "))  except n < 2:  print("Error (n < 2)!")  else:  print("The prime factors of",str(n),"are:")  while factor <= n:  if n % factor == 0:  n /= factor  print(factor)  else:  factor += 1  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 2 бали |
| 27. | Умова:  Виведіть на екран символи таблиці ASCII від 0 до 127 разом з їх номером у десятковій, двійковій, вісімковій та шістнадцятковій системах числення.  Код: import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  print("Десяткова ","%8s" % "Символ","%10s" % "Двійкова","%10s" % "Вiсімкова","%15s" % "Шістнадцяткова")  for i in range(128):  print("%5s" % i, "%10s" % chr(i), "%12s" % bin(i), "%9s" % oct(i), "%10s" % hex(i))  printTimeStamp("Віталій Дудник та Коробова Даніїла")  Скріншот: | 2 бали |
| 28. | Обчислити ітогові дані по роках, щорічні та щомісячні середні значення опадів за період у декілька років | 2 бали |

**Рівень С**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Умова:  Рулетка має 38 комірок: 18 чорних, 18 червоних і 2 зелених. Зелені комірки позначаються як 0 та 00. Червоними комірками є 1, 3, 5, 7, 9, 12, 14, 16, 18, 19, 21, 23, 25, 27, 30 32, 34 та 36. Решта цілих чисел з діапазону від 1 до 36 є чорними.  На рулетку можна зробити багато ставок. Для цієї вправи розглянемо наступні можливі ставки:  • Одне число (від 1 до 36, 0 або 00)  • Червоне vs Чорне  • Парне vs Непарне (0 та 00 не включаються)  • Від 1 до 18 vs від 19 до 36  Напишіть програму, яка симулює кручення колеса рулетки за допомогою генератора випадкових чисел. Виведіть вибране число та всі ставки, які потрібно оплатити. Наприклад, для 13:  На рулетці випало 13...  Виплатити 13  Виплатити Black  Виплатити Odd  Виплатити 1 to 18  Якщо результати симуляції випадають на 0 або 00, слід вивести Виплатити 0 або Виплатити 00 без іншої інформації.  Код:  import datetime,random  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  red = ["1","3","5","7","9","12","14","16","18","19","21","23','25",'27','30',"32","34","36"]  random\_numb = random.randint(0,37)  if random\_numb == 37:  random\_numb = "00"  print("На рулетці випало",random\_numb,"\n"+"Виплатити",random\_numb)  if str(random\_numb) in "00":  print("Виплатити зелене")  elif str(random\_numb) in red:  print("Виплатити червоне")  else:  print("Виплатити чорне")  if str(random\_numb) == "0" and "00":  pass  elif int(random\_numb) % 2 == 0:  print("Виплатити парні")  else:  print("Виплатити непарні")  if 0 < random\_numb < 19:  print("Виплатити від 1 до 18")  elif 18 < random\_numb < 37:  print("Виплатити від 19 до 36")  else:  pass  printTimeStamp("Віталій Дудник та Коробов Даніїл")  Скріншот: | 4 бали |
| 2. | Умова:  Напишіть програму, яка зчитує дату від користувача та обчислює дату наступного дня. Дата буде вводитись у числовій формі трьома окремими виразами (рік, місяць, день). Наприклад, якщо користувач вводить значення, яке представляє день 2017-05-10, Ваша програма має показати повідомлення, яке говоритиме, що наступним днем буде 2017-05-11.   * Якщо користувач вводить значення на зразок 2017-04-30, програма має вказати, що наступним днем буде 2017-05-01. * Якщо користувач вводить значення, що представляє дату 2016-12-31, програма має вивести наступний день - 2017-01-01. * Забезпечте коректну роботу для високосних років.   Не використовуйте вбудованих методів для обчислення наступного дня!  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  neadekvat=1  while neadekvat ==1:  neadekvat-=1  year = int(input("Введіть рік: "))  month = int(input("Введіть місяць: "))  day = int(input("Введіть день: "))  feva= int(str(year)[-2:])  if (month or year or day) <= 0 or (month==2 and year%4==0 and day>=29):  print("Ти неaдаекват!!!")  neadekvat+=1  break  if (day == 31 and (month == 1 or 3 or 5 or 7 or 8 or 10 or 12)) or (day== 30 and (month == 2 or 4 or 6 or 9 or 11)) or (day==28 and month== 2 and feva%4 !=0) or (day==29 and month== 2 and feva%4 ==0):  day= 1  year =year if month != 12 else year+1  month=1 if month==12 else month+1  else:  day+=1  print("Наступний день: {}-{}-{}".format(year, (month if month>9 else "0"+str(month)), (day if day>9 else "0"+str(day))))  printTimeStamp("Коробов Даніїл та Віталій Дудник")  Скріншот: | 5 балів |
| 3. | Умова:  Припустимо, що в нас є показники температури в кімнаті та знадвору. Програма повинна вирішувати, яку техніку краще ввімкнути: кондиціонер, термостат чи обігрівач. Температура в приміщенні поступово змінюється, що залежить від різниці температур в кімнаті і знадвору , а також коефіцієнту пропорційності :  На вхід користувач вводить температуру в кімнаті та на вулиці.   * Якщо температура в кімнаті знаходиться в комфортних межах (від 22°С до 24°С), вмикається термостат, який буде підтримувати таку температуру (коефіцієнт пропорційності ). Проблема в тому, що він може працювати підряд лише 8 годин. * Якщо в кімнаті температура нижча за комфортну, потрібно вмикати обігрівач (коефіцієнт пропорційності ). * Якщо в кімнаті температура вища за комфортну, необхідно увімкнути кондиціонер (коефіцієнт пропорційності ). * Якщо відключені всі пристрої (температура комфортна, проте термостат «відпочиває»), відбувається природне охолодження (коеф. ).   Програма має виводити на екран час, поточну температуру в кімнаті та стан приладів (true/false) протягом 24 годин. Вважатимемо, що температура надворі береться опівночі і з 5-ї до 15-ї години росте на 1 градус з кожною годиною, а в решту часу – зменшується на 1 градус із кожною годиною.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  s\_temperature= int(input("Введіть температуру на вулиці: "))  r\_temperature= int(input("Введіть температуру в кімнаті: "))  chasiki=0  k=0  termostat=True  while chasiki!= 24:  if 5<=chasiki<=15:  s\_temperature+=1  else:  s\_temperature-=1  if 22<=r\_temperature<=24 and termostat==True:  termostat= False  k=0  for i in range(8):  print("Температура в кімнаті - {:.2f}`C. Година - {}:00. Термостат працює.".format(r\_temperature,chasiki if chasiki> 9 else "0"+str(chasiki)))  r\_temperature= r\_temperature+k\*abs((r\_temperature-s\_temperature))  if 5 <= chasiki <= 15:  s\_temperature += 1  else:  s\_temperature -= 1  chasiki+=1  elif 22<=r\_temperature<=24 and termostat==False:  k=-0.02  r\_temperature = r\_temperature + k \* abs((r\_temperature - s\_temperature))  chasiki+=1  print("Температура в кімнаті - {:.2f}`C. Година - {}:00. Всі прилади вимкнені.".format(r\_temperature,chasiki if chasiki > 9 else "0" + str(chasiki)))  elif r\_temperature<22:  k=0.11  r\_temperature = r\_temperature + k \* abs((r\_temperature - s\_temperature))  chasiki+=1  print("Температура в кімнаті - {:.2f}`C. Година - {}:00. Обігрівач працює.".format(r\_temperature,chasiki if chasiki > 9 else "0" + str(chasiki)))  elif r\_temperature>24:  k=-0.09  r\_temperature = r\_temperature + k \* abs((r\_temperature - s\_temperature))  chasiki+=1  print("Температура в кімнаті - {:.2f}`C. Година - {}:00. Кондіціонер працює.".format(r\_temperature,chasiki if chasiki > 9 else "0" + str(chasiki)))  chasiki+=1  printTimeStamp("Коробов Даніїл та Віталій Дудник")  Скріншот: | 5 балів |
| 4. | Умова:  Інтернет-провайдер має три тарифи для населення:   |  |  | | --- | --- | | Тариф 1000 | 1000 Мб на місяць за 20 грн, позатарифно: 1Мб = 0.05 грн. | | Тариф 2000 | 2000 Мб на місяць за 35 грн, позатарифно: 1Мб = 0.04 грн. | | Тариф 5000 | 5000 Мб на місяць за 85 грн., позатарифно: 1 Мб = 0.02 грн. |   Напишіть програму, яка повинна отримувати цифру-тариф та кількість витрачених мегабайтів, а виводити підсумковий рахунок за місяць. Якщо використовується тариф 1000 або 2000, виведіть додатково інформацію про те, скільки б платили користувачі, якби перейшли на більш дорогі тарифи.  Код:  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  tariff = input("Введіть цифру тарифу яким користуєтесь:")  megabytes = int(input("Введіть скільки мегабайтів використали:"))  if tariff == "1000":  print("Сейчас вы платите {} за {} Мб, а с Тарифом 2000 - {}, а с Тарифом 5000 - {}".format(((((megabytes- 1000) \* 0.05)if megabytes>=1000 else 0)+20),  megabytes, ((((megabytes- 2000) \* 0.04)+35) if megabytes> 2000 else 35),((((megabytes- 5000) \* 0.02)+85) if megabytes> 5000 else 85) ))  elif tariff == "2000":  print("Сейчас вы платите {} за {} Мб, а с Тарифом 5000 - {} ".format((((((megabytes- 2000) \* 0.04)if megabytes>=2000 else 0)+35)), megabytes,((((megabytes- 5000) \* 0.02)+85) if megabytes> 5000 else 85)))  elif tariff == "5000":  print("Сейчас вы платите {} за {} Мб".format(((((megabytes- 5000)if megabytes>=5000 else 0) \* 0.02)+85), megabytes))  printTimeStamp("Коробов Даніїл та Віталій Дудник")  Скріншот: | 4 бали |
| 5. | Умова:  Для гри в зонк потрібно 6 шестигранних кубиків, які будуть кидатись одночасно. Доступні наступні комбінації:    Напишіть підпрограму, яка аналізує кидок одного гравця. При цьому генерується 6 випадкових чисел (від 1 до 6). Виведіть на екран доступні комбінації за результатом кидка та очки, які за них набере користувач.  Код:  import random, datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  points = 0  attempt =1 # Попытка, добавлеться если три пары или шесть одинаковых  while attempt == 1:  attempt -= 1  bone= [random.randint(1,6) for i in range(6)]  print(bone)  break\_group = 0 #Ввел для того что бы если уже нашло 3 или больше одинаковых костей (1\*), то не стработали условие "с тремя парми" (2\*)  for i in range(1,7):  if bone.count(i) >= 3:  print("{} однакові кості '{}' +{}".format(bone.count(i), i,(bone.count(i) - 2) \* 100 \*(10 if i == 1 else i)))  points += (bone.count(i) - 2) \* 100 \*(10 if i == 1 else i)  break\_group +=1 #(1\*)  if len(set(bone))==3 and break\_group == 0: # (2\*) Тут считаеться когда у тебя три пары костей(сет убирает все одинаковые элементы, а потом считает количество), но может быть [1,1,1,1,2,3] и он посчитает что там 3 пары, для этго и ввел break\_group = 0  print("Три пари +750 та ще одна спроба")  points += 750  attempt += 1  elif break\_group==0:  points += bone.count(1) \* 100  points += bone.count(5) \* 50  if len(set(bone)) == 6:  print("Три пари +1500 та ще одна спроба")  points += 1500  attempt += 1  print("Ваші очки:", points)  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 5 балів |
| 6. | Умова:  Нижче наведена таблиця описує октави музичних нот, починаючи з middle C (С4) разом з їх частотами.   |  |  | | --- | --- | | **Нота** | **Частота (Гц)** | | C4 | 261,63 | | D4 | 293,66 | | E4 | 329,63 | | F4 | 349,23 | | G4 | 392,00 | | A4 | 440,00 | | B4 | 493,88 |   Програма має коректно виводити частоти для нот всіх октав – від C0 до C8. Для того, щоб їх знайти, потрібно враховувати наступне правило: частота ноти в октаві n – половина частоти ноти в октаві n+1. Додаток має підтримувати 2 варіанти роботи: 1) На вхід програми має надходити нота відповідної октави, а на виході виводитись її частота.  2) За введеною частотою визначити ноту.  *Підказка: для виконання завдання потрібно виділити окремо символи з двосимвольної назви ноти. Далі можна визначити частоту ноти в четвертій октаві, а потім поділити її на 24−x, де x – номер октави, введеної користувачем.*  *Код:*  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  fourth\_octave= {"C": 261.63, "D": 293.66, "E": 329.63, "F": 349.23, "G": 392.00, "A": 440.00, "B": 493.88}  x= input("Ведіть ноту, щоб узнати її частоту, або частоту, щоб знайти ноту: ")  try:  note = x[0]  octave = int(x[1])  note\_hz = fourth\_octave[note] / 2\*\*(4-octave)  print("Частота ноти {}: {} Гц".format(x, note\_hz))  except KeyError:  x= float(x)  stop=0  start\_octaves = 4  iter= 0  while -1< start\_octaves < 9:  if stop == -1:  break  if iter > 56:  print("Нота поза діапазону")  break  if x > 493.88:  x=x/2  start\_octaves += 1  if x < 261.63:  x = x \* 2  start\_octaves -= 1  for i in fourth\_octave:  if x == fourth\_octave[i]:  print("Це частота ноти:", i+str(start\_octaves))  stop= -1  iter+=1  else:  print("Нота поза діапазону")  printTimeStamp("Даніїл Коробов та Дудник Віталій")  Скріншот: | 4 бали |

**Зведена таблиця дня 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Рівень А** | **Бали** | **Рівень В** | **Бали** | **Рівень С** | **Бали** |  |
| **1А1** | **1** | **1В1** | **2** | **1С1** | **4** |  |
| **1А2** | **0,5** | **1В2** | **1,5** | **1С2** | **5** |  |
| **1А3** | **1** | **1В3** | **2** | **1С3** | **5** |  |
| **1А4** | **1** | **1В4** | **2,5** | **1С4** | **4** |  |
| **1А5** | **0,5** | **1В5** | **2,5** | **1С5** | **5** |  |
| **1А6** | **1** | **1В6** | **2,5** | **1С6** | **4** |  |
| **1А7** | **1,5** | **1В7** | **2,5** |  |  |  |
| **1А8** | **1** | **1В8** | **2,5** |  |  |  |
| **1А9** | **1** | **1В9** | **2,5** |  |  |  |
| **1А10** | **1** | **1В10** | **3** |  |  |  |
| **1А11** | **1** | **1В11** | **2** |  |  |  |
| **1А12** | **1** | **1В12** | **2,5** |  |  |  |
| **1А13** | **1** | **1В13** | **2** |  |  |  |
| **1А14** | **1** | **1В14** | **1,5** |  |  |  |
| **1А15** | **0,5** | **1В15** | **1,5** |  |  |  |
| **1А16** | **0,5** | **1В16** | **2,5** |  |  |  |
| **1А17** | **0,5** | **1В17** | **2,5** |  |  |  |
| **1А18** | **1,5** | **1В18** | **2,5** |  |  |  |
| **1А19** | **1** | **1В19** | **2** |  |  |  |
| **1А20** | **1,5** | **1В20** | **2** |  |  | **Кількість** |
| **1А21** | **1,5** | **1В21** | **2** |  |  | **55** |
|  |  | **1В22** | **2** |  |  |  |
|  |  | **1В23** | **2,5** |  |  |  |
|  |  | **1В24** | **2** |  |  | **Сума** |
|  |  | **1В25** | **2** |  |  | **109** |
|  |  | **1В26** | **2** |  |  |  |
|  |  | **1В27** | **2** |  |  |  |
|  |  | **1В28** | **2** |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Сума А** | **20** | **Сума В** | **62** | **Сума С** | **27** |  |

**День 02. Використання вбудованих структур даних**

**Рівень А**

|  |  |  |
| --- | --- | --- |
| **№** |  | **Бали /**  **Зв’язки** |
| 1. | **УМОВА:**  При аналізі даних, зібраних у результаті експерименту, часто може виникати потреба у видаленні найбільш екстремальних значень до виконння обчислень. Ваша програма має видалити зі списку n (ціле додатнє число, введене користувачем), по троє найбільших та найменших значень і вивести решту значень (порядок не важливий). Також забезпечте коректну обробку ситуації, коли у списку не вистачатиме елементів для видалення.  **КОД:**  import datetime  def printTimeStamp(name):   print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  n =[1,2,3,4,5,6,7,8,9,10] print(n) delete = int(input("Введіть число для видалення: ")) n.remove(delete) if len(n) > 5:  for i in range(3):  n.remove(max(n))  n.remove(min(n)) else:  print("В списку недостатньо елементів") print(n) printTimeStamp("Віталій Дудник, Коробов Даніїл")  **ВИВІД:** | 1 бал |
| 2. | Напишіть програму, яка зчитує числа від користувача, поки не буде введено порожній рядок. Далі вона повинна знайти середнє значення із введених чисел та вивести 2 списки: введених чисел, що не перевищують середнє значення та, відповідно, більше або рівні за нього.  **КОД:**  import datetime  def printTimeStamp(name):   print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  listok1 = [] while True:  number = input("Введіть число:")  if number == "":  break  elif not number.isdigit():  print("Це не число")  else:  listok1.append(int(number)) average = sum(listok1) / len(listok1) listok2 = [] listok3 = [] for i in listok1:  if i < average:  listok2.append(i)  else:  listok3.append(i)  print("1:", listok2) print("2:", listok3) printTimeStamp("Віталій Дудник, Коробов Даніїл")  **ВИВІД:** | 1 бал |
| 3. | Напишіть програму, яка зчитує від користувача слова, кожне з нового рядка. Ввід закінчується порожнім рядком, після чого виводиться список всіх унікальних слів, які були введені. Наприклад, для вводу  first  second  first  third  second  програма повинна буде вивести  first  second  third  **КОД:**  import datetime  def printTimeStamp(name):   print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) words = set() while True:  word = input("Введіть слово:")  if word == "":  break  else:  words.add(word) for i in words:  print(i) printTimeStamp("Віталій Дудник, Коробов Даніїл")  **ВИВІД:** | 1 бал |
| 4. | Напишіть додаток, який використовує генерування випадкових чисел для створення речень. Використовуйте 4 списки рядків: article, noun, verb та preposition. Створіть речення, обравши з кожного масиву слова в такому порядку: артикль, іменник, дієслово, прийменник, артикль та іменник. Після вибору кожного слова виконайте його конкатенацію з реченням. Слова мають відокремлюватись пробілами. Загальне речення має починатись з великої літери та закінчуватись крапкою. Додаток має згенерувати та відобразити 10 речень.  *Артиклі:* "the", "a", "one", "some" та "any";  *Іменники:* "boy", "girl", "dog", "town" та "car";  *Дієслова*: "drove", "jumped", "ran", "walked" та "skipped";  *Прийменники*: "to", "from", "over", "under" та "on".  **КОД:**  import datetime from random import randint  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) article = ["the", "a", "one", "some", "any"] noun = ["boy", "girl", "dog", "town", "car"] verb = ["drove", "jumped", "ran", "walked", "skipped"] preposition = ["to", "from", "over", "under", "on"] listok = [article, noun, verb, preposition, article, noun] for i in range(10):  sentence = []  for j in listok:  sentence.append(j[randint(0,len(j)-1)])  sentence[0] = sentence[0].capitalize()  print(str(i+1)+":"," ".join(sentence)) printTimeStamp("Віталій Дудник, Коробов Даніїл")  **ВИВІД:** | 1,5 бали |
| 5. | Нехай грає музика! Запрограмуйте комп’ютер на відтворення мелодії, заданої одновимірним масивом частот та одновимірним масивом тривалостей відтворення відповідної частоти. Відповідні пари значень наведені нижче:  (659,250), (659,250), (659,300), (523,250), (659,250), (784,300), (392,300), (523,275), (392,275), (330,275), (440,250), (494,250), (466,275), (440,275), (392,275), (659,250), (784,250), (880,275), (698,275), (784,225), (659,250), (523,250), (587,225), (494,225).  Для підключення біперу потрібно імпортувати модуль  import winsound  і викликати функцію  winsound.Beep(частота у Гц, тривалість у мс)  **КОД:**  import winsound, datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) arrey = [(659,250), (659,250), (659,300), (523,250), (659,250), (784,300),  (392,300), (523,275), (392,275), (330,275), (440,250), (494,250),  (466,275), (440,275), (392,275), (659,250), (784,250), (880,275),  (698,275), (784,225), (659,250), (523,250), (587,225), (494,225)] frequency = [i[0] for i in arrey] duration = [i[1] for i in arrey] i = 0 while i != len(arrey):  winsound.Beep(frequency[i], duration[i])  i += 1 printTimeStamp("Віталій Дудник, Коробов Даніїл") | 1 бал |
| 6. | **УМОВА**:  Якщо Вам сподобалась попередня задача, спробуйте тепер іншу мелодію, збережену в *двовимірному* масиві (списку списків):  (440,500), (440,500), (440,500), (349,350), (523,150), (440,500), (349,350), (523,150), (440,1000), (659,500), (659,500), (659,500), (698,350), (523,150), (415,500), (349,350), (523,150), (440,1000)  **КОД:**  import winsound, datetime  def printTimeStamp(name):   print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  l = [(440,500), (440,500), (440,500), (349,350), (523,150), (440,500), (349,350), (523,150), (440,1000), (659,500), (659,500), (659,500), (698,350), (523,150), (415,500), (349,350), (523,150), (440,1000)] for i in l:  winsound.Beep(i[0], i[1])   printTimeStamp("Коробов Даніїл, Дудник Віталій") | 1 бал |
| 7. | **УМОВА:**  Звичайна колода карт має 52 карти, кожна з яких описується мастю і значенням. Мастями можуть бути піки (spades), черви (hearts), трефи (diamonds) та бубни (clubs), а значеннями – числа від 2 до 10, валет (Jack), дама (Queen), король (King) і туз (Ace). Приклади позначень карт описані в таблиці.    Напишіть програму, яка буде створювати колоду карт у вигляді списку.  **КОД:**  import itertools, datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  vals = ['2', '3', '4', '5', '6', '7', '8', '9', '10', 'jack', 'queen', 'king', 'ace'] suits = ['spade', 'club', 'heart', 'diamond'] deck = list(itertools.product(vals, suits))  for i in deck:  print("{} of {}".format(i[0],i[1]))  printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 8. | Алгоритми тасування колоди карт, побудованих на базі алгоритму Фішера-Йетса описані [тут](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D1%81%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%A4%D0%B8%D1%88%D0%B5%D1%80%D0%B0_%E2%80%94_%D0%99%D0%B5%D1%82%D1%81%D0%B0). Створіть колоду карт, яку можна буде потасувати за допомогою алгоритму Сатолло. Виведіть початкову колоду та перетасовану на екран.  **КОД:**  import itertools, datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) from random import randrange  vals = ['2', '3', '4', '5', '6', '7', '8', '9', '10', 'jack', 'queen', 'king', 'ace'] suits = ['spade', 'club', 'heart', 'diamond']  deck = list(itertools.product(vals, suits)) print("STANDARD DECK:") for i in deck:  print("{} of {}".format(i[0],i[1]))  def sattoloCycle(items):  i = len(items)  while i > 1:  i = i - 1  j = randrange(i)  items[j], items[i] = items[i], items[j]  return sattoloCycle(deck) print("NEW DECK:") for i in deck:  print("{} of {}".format(i[0],i[1]))  printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 9. | На базі попередньої задачі перетасуйте колоду також за допомогою алгоритмів Дуршенфельда та Фішера-Йетса. | 1,5 бали |
| 10. | Створивши та перетасувавши колоду в попередніх задачах, прийшов час здати їх гравцям. Кількість гравців та карт для кожного з них залежить від гри, тому користувач повинен після перетасовки ввести кількість карт на одного гравця та кількість гравців. За цими даними потрібно вивести інформацію зі списку про гравців та карти, що їм були здані. Забезпечте коректну роботу програми, коли карт для гравців потрібно більше, ніж є в колоді. | 1 бал |
| 11. | Напишіть програму, яка симулюватиме 1000 кидків пари костей та виводитиме статистику сум, які випадають у відсотках. Оформити результат можна так:    Очікуваний відсоток – стандартне значення, отримане з розподілу Гауса.  **КОД**:  import random import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) points =[] for i in range(1000):  points.append(random.randint(2,12)) print(" 2: {}%\n 3: {}%\n 4: {}%\n 5: {}%\n 6: {}%\n 7: {}%\n 8: {}%\n 9: {}%\n 10: {}%\n 11: {}%\n 12: {}% "  .format(points.count(2)/10, points.count(3)/10, points.count(4)/10, points.count(5)/10, points.count(6)/10, points.count(7)/10,  points.count(8)/10, points.count(9)/10, points.count(10)/10, points.count(11)/10, points.count(12)/10, )) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |
|  |  |  |
|  |  |  |

**Рівень В**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Канадський поштовий код складається з 6 символів: перший, третій і п’ятий з них букви, а решта – цифри. Провінцію можна визначити за першим символом коду (таблиця). Букви D, F, I, O, Q, U, W та Z не можуть бути першими.    Другий символ позначає сільську місцевість (0) або місто (інакше).  Напишіть програму, яка зчитує поштовий код та визначає провінцію та місцевість проживання. Наприклад, T2N1N4 вказує на проживання в місті провінції Альберта. Якщо введено X0A1B2, поштовий код вказує на село в Nunavut або Northwest Territories. Використовуйте словник для відображення перших символів коду в назву провінції. Забезпечте коректну роботу програми для некоректного коду.  **КОД:**  import re, datetime  def printTimeStamp(name):   print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  province = {"A": "Newfoundland", "B": "Nove Scotia", "C": "Prince Edward Island",  "S": "Saskatchewan", "T": "Albera", "V": "British Columbia",  "X": "Nunavut or Northwest Territories", "Y": "Yukon","R":"Manitoba",  "E": "New Brunswick", "G":"Quebec", "H":"Quebec", "J":"Quebec",  "K":"Ontario", "L":"Ontario", "M":"Ontario", "N":"Ontario", "P":"Ontario"} code = input("Enter mail-code: ") print(code) pattern = "([A-Z][0-9]){3}" if re.match(pattern, code):  if code[0] in province:  print("Province:", province.get(code[0]))  else:  print("Uncorrect mail-code. D, F, I, O, Q, U, W and Z can not be first letter")  if code[1] == 0:  print("Type: Village")  else:  print("Type: City") else:  print("Uncorrect mail-code") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |
| 2. | На більшості кнопочних телефонів для набору текстового повідомлення використовують цифрову клавіатуру. Оскільки кожна кнопка прив’язана до кількох букв, її потрібно натискати декілька разів для вибору потрібної відповідно до таблиці. Натиснення цифри 2, 3, 4 або 5 разів генерує другу, третю, четверту або п’яту букву для цієї кнопки.   |  |  | | --- | --- | | **Кнопка** | **Символи** | | 1 | . , ? ! : | | 2 | A B C | | 3 | D E F | | 4 | G H I | | 5 | J K L | | 6 | M N O | | 7 | P Q R S | | 8 | T U V | | 9 | W X Y Z | | 0 | пробіл |   Напишіть програму, яка показує, скільки та яких натиснень кнопок потрібно, щоб набрати текст користувача. Сконструюйте словник, який відображає кожну букву чи символ на кнопку, яку потрібно натиснути. Наприклад, для тексту Hello, World! програма має вивести 4433555555666110966677755531111. Забезпечте коректну роботу як з малими, так і великими літерами. Ігноруйте символи, не представлені в таблиці.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) dictionary = {1: [".", ",", "?", "!", ":"], 2: ["A", "B", "C"],  3: ["D", "E", "F"], 4: ["G", "H", "I"],  5: ["J", "K", "L"], 6: ["M", "N", "O"],  7: ["P", "Q", "R", "S"], 8: ["T", "U", "V"],  9: ["W", "X", "Y", "Z"], 10: [" "]} text = (input("Text:")).upper() for i in text:  for key, values in dictionary.items():  for value in values:  if value == i:  print(str(key)\*(values.index(i)+1), end="") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 3 бали |
| 3. | Токенізація – процес конвертації рядка в набір підрядків (substring), які називають токенами. У багатьох випадках з набором токенів працювати набагато простіше, ніж з початковим рядком.  У математичних виразах токенами виступають оператори, числа та дужки. Деякі токени просто визначити, наприклад, \*, /, ˆ, ( та ), оскільки вони односимвольні і не можуть бути частиною іншого токену. Символи + та – обробляти складніше, тому що вони можуть представляти оператор додавання/віднімання або бути частиною токену-числа.  *Підказка: Символ + або – є оператором, якщо непробільний символ після нього є частиною токену-числа або перший непробільний перед ним є дужкою. Інакше ці символи є частинами числа.*  Напишіть програму, яка обробляє введений користувачем математичний вираз та виводить список його токенів з їх типом. Вважаємо, що токенами є дужки, оператор, число з опційним знаком + або - (для простоти братимемо тільки цілі числа.  **КОД:**  import datetime, re def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  expression = input("Введіть вираз для токенізації: ") tokens = [i for i in expression.replace(" ","")] tokens1 = [] for i in range(len(tokens)-1):  try :  while True:  if tokens[i].isdigit() and tokens[i+1].isdigit():  tokens[i] = tokens[i] + tokens[i+1]  tokens.remove(tokens[i+1])  else:  break  except IndexError:  pass i = 0 while i != len(tokens):  item = tokens[i]  if item in "\*/^()":  tokens1.append(item)  elif item in "+-":  if tokens[i-1] not in "/\*^()" and re.match("[0-9]",tokens[i+1][0]):  tokens1.append(item)  else:  tokens1.append(item)  tokens1.append(tokens[tokens.index(item) + 1] + tokens[tokens.index(item) + 2])  i += 2  elif item.isdigit():  tokens1.append(item)  i += 1 for i in tokens1:  if i.isdigit():  print(i, "- число" )  elif i in "/\*^+-":  print(i, "- оператор")  elif i == "()":  print(i, "- дужка")  elif i[0] in "+-" and i[1].isdigit():  print(i, "- число з опційним знаком") printTimeStamp("Дудник Віталій")  **ВИВІД:** | 3 бали |
| 4. | Решето Ератосфена – алгоритм, який дозволяє знайти прості числа, що не перевищують деякого n. Він був придуманий понад 2000 років тому і описується так:  Запишіть всі числа від 0 до n  Викресліть 0 і 1, тому що вони не є простими  Задайте p = 2  While p < n do  Викреслити всі числа, кратні p (проте не сам p)  Перейти (змінна p) до наступного числа в списку, яке  не буде закресленим  Вивести всі незакреслені числа.  Сформуйте список усіх чисел від 2 до введеного значення n, а потім видаляйте з нього всі числа, які викреслює алгоритм. У результаті повинен отриматись список із простих чисел для введеного діапазону.  Підказка. Пришвидшити роботу початкового алгоритму можна за рахунок неповного перебору чисел. Насправді, достатньо перевірити числа не до n, а лише до .  **КОД:**  import datetime  def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) n = int(input("Введіть число n: ")) listok = [i for i in range(n+1) if i > 1] p = 2 while p\*\*2 < n:  for i in listok:  if p == i:  continue  elif i % p == 0:  listok.remove(i)  try:  p = listok[listok.index(p)+1]  except IndexError:  break listok = [str(i) for i in listok] print(", ".join(listok)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |
| 5. | У грі Scrabble™ кожна буква оцінюється балами, як показано в таблиці. Загальна оцінка слова – це сумарна кількість очків за кожну букву:    Напишіть програму, яка обчислює та виводить Scrabble™-оцінку для слова. Створіть словник, який відображає букви на бальні значення, а потім з його допомогою аналізуйте введені слова, виводячи їх оцінку в балах.  У грі Scrabble™ включено деякі squares, які домножають значення букви чи всього слова на певні значення. Це правило в задачі ігнорується.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) dictionary = {1: ["A", "E", "I", "L", "N", "O", "R", "S", "T", "U"],  2: ["D", "G"], 3:["B", "C", "M", "P"], 5: ["K"],  4: ["F", "H", "V", "W", "Y"], 8: ["J", "X"], 10:["Q", "Z"]} words = input("Word(s): ") words = words.split(" ") for word in words:  result = 0  for letter in word:  for key, values in dictionary.items():  for value in values:  if value == letter.upper():  result += key  print(word,"-",result) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 3 бали |
| 6. | Багато фінансових документів передбачають запис суми грошей як у числовій, так і в текстовій формі. Напишіть програму, яка буде конвертувати суми від 0 до 10 000 грн. у текстове представлення. Наприклад, 3726 грн – три тисячі сімсот двадцять шість гривень. Використовуйте списки для переліку можливих текстових представлень для одиниць, десятків, сотень та тисяч гривень.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) one = ["одна", "дві", "три", "чотири", "п'ять", "шість", "сім", "вісім", "дев'ять"] two\_one = ["одинадцять", "дванадцять", "тринадцять", "чотирнадцять", "п'ятнадцять",  "шістнадцять", "сімнадцять", "вісімнадцять", "дев'ятнадять"] two = ["","двадцять", "тридцять", "сорок", "п'ятдесят", "шістдесят", "сімдесят", "вісімдесят", "д'евяносто"] # в списку two перший елемент пустий для правильного індексу three = ["сто", "двісті", "триста", "чотириста", "п'ятсот", "шістсот", "сімсот","вісімсот", "дев'ятсот"] four = ["одна тисяча", "дві тисячі", "три тисячі","чотири тисячі", "п'ять тисяч",  "шість тисяч", "сім тисяч", "вісім тисяч", "дев'ять тисяч"] five = ["десять тисяч"] dictionary = {0:one, 1:two, 2:three, 3:four, 4:five} while True:  n = input("Число для конвертації: ")  if n.isdigit():  if 0 < int(n) <= 10000:  break  else:  print("Число не в діапазоні від 0 до 10 000")  else:  print("Це не число") n = str(n) grn = [] for i in range(len(n)):  if int(n[-1-i]) != 0:  if i == 0 and n[-2] == 1:  word = two\_one[int(n[-1])-1]  grn.append(word)  print(i, word)  elif i == 1 and n[-2] == 1:  continue  else:  word = dictionary.get(i)[int(n[-1-i])-1]  grn.append(word)  print(i, word)  else:  continue grn.reverse() print(grn, "грн") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2,5 бали |
| 7. | Припустимо, на залізничному вокзалі присутня автоматизована система з управління камер зберігання для багажу пасажирів. Інформація про камери зберігається у вигляді переліку «Прізвище пасажира-час здачі багажу-час видачі багажу». На вокзалі присутні 12 комірок, пасажиру відкривається вільна комірка з мінімальним номером.  Вважаємо, що робота програми обмежується одним днем, а час здачі та видачі багажу вводить сам пасажир. Для роботи з програмою спочатку треба ввести кількість пасажирів, а потім їх дані, наприклад, так:  3  Іваненко 09:45 12:00  Петренко 10:00 11:00  Сидоренко 12:00 13:12  На екран має бути виведено, яку комірку відкривати кожному з пасажирів.  Іваненко 1  Петренко 2  Сидоренко 1 | 3 бали |
| 8. | Напишіть функцію precedence(), яка повертає ціле число, що показує пріоритет математичного оператора. Рядок, який містить оператор, передається як єдиний параметр функції. Сама функція має повертати 1 для + та -, 2 для \* та /, 3 для ˆ. Якщо переданий рядок не має цих операторів, функція має повернути -1.  Основна програма буде зчитувати оператор від користувача та або виводить пріоритет оператора, або повідомлення, що не було введено оператор. Your main program should only run when the file containing your solution has not been imported into another program.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) oper = input("Введіть оператор: ") def precedence(operator):  if operator in ["-","+"]:  return 1  if operator in ["\*","/"]:  return 2  if operator == "^":  return 3  else:  return -1  if precedence(oper) >= 1:  print("Пріорітет операції:",precedence(oper)) elif precedence(oper)== -1:  print("Невірні дані!") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |
| 9. | Створіть плейліст, кожна пісня якого має назву, виконавця, альбом та тривалість. Плейліст дозволяє додавати та видаляти пісню, отримувати розмір та очищати його, а також виконувати форматований вивід композицій і повну тривалість його відтворення. | 3 бали |
| 10. | Напишіть програму, яка зчитує IPv4 адресу та перевіряє її валідність. Форма запису IPv4 адреси – x.x.x.x, де кожен x має бути цілим числом з діапазону [0, 255].  **КОД:**  import re, datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) ip = input("Введіть свою IPv4 адресу: ") val= re.split("\.", ip) proverka=0 if len(val) != 4:  print("IPv4 адреса неправильна!") else:  for i in val:  if int(i) >255 or int(i)<0:  print("IPv4 адреса неправильна!")  proverka+=1  break  if proverka==0:  print("Адреса валідна") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |

|  |  |  |
| --- | --- | --- |
| 11. | Введіть з клавіатури деякий текст та підрахуйте в ньому кількість слів, букв, цифр,малих літер, великих літер, пробілів.  **КОД:**  import re, datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) points =[] text = input("Enter text: ") words = re.split(" ", text) vowels= re.findall("e|i|u|o|a", text.lower()) consonants =re.findall("B|C|D|F|G|J|K|L|M|N|P|Q|R|S|T|V|W|X|Y|Z", text.upper()) big =re.findall("[A-Z]", text) small =re.findall("[a-z]", text) space =re.findall(" ", text) letters =re.findall("[A-Za-z]", text) numbers =re.findall("\d", text) print(words) print("Words: {} Letters: {} Number: {} Space: {} Big letters: {} Small letters: {} Vowels: {} Consonants: {}".format  (len(words),len(letters),len(numbers), len(space), len(big), len(small), len(vowels), len(consonants))) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |

|  |  |  |
| --- | --- | --- |
| 12. | Невелика авіакомпанія потребує систему автоматизації резервування місць у літаку на 12 місць. Ваша програма має пропонувати наступні альтернативи:  Натисніть 1 для першого класу (місця 1-6)  Натисніть 2 для економного класу (місця 7-12).  Потім Ваш застосунок повинен відображати посадкові місця, вказуючи на замовлене місце та його клас.  Використовуйте список булевих значень для представлення посадкових місць у літаку. Ваш застосунок має ніколи не давати резервувати вже зарезервовані місця, а повинен видавати перше вільне місце в класі. Коли місця з економного класу закінчились, програма повинна запропонувати квиток першого класу. Якщо таке не підходить, програма виведе повідомлення «Наступний виліт через 3 години».  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) reserv = {1:True, 2:True, 3:True, 4:True, 5:True, 6:True, 7:True, 8:True, 9:True, 10:True, 11:True, 12:True} place= 0 def business():  global place  for i in range(1, 7):  if reserv[i] == True:  print("Ваше посадкове місце {}. Бізнес-клас".format(i))  reserv[i] = False  place += 1  break  else:  print("Місця бызнес-класу закінчились!")  pic= int(input("Натисніть '1' якщо хочете зарезервувати місця в економ-класі/ Натисніть '2' для відміни: "))  if pic == 1:  economy()  else:  print("Наступний виліт через 3 години")  place=12 def economy():  global place  for i in range(7,13):  if reserv[i]== True:  print("Ваше посадкове місце {}. Економ-клас".format(i))  reserv[i] = False  place += 1  break  else:  print("Місця економ-класу закінчились!")  pic = int(input("Натисніть '1' якщо хочете зарезервувати місця в бізнес-класі/ Натисніть '2' для відміни: "))  if pic == 1:  business()  else:  print("Наступний виліт через 3 години")  place = 12 while place != 12:  num = int(input("Натисніть '1' для першого класу/ Натисніть '2' для економного класу/ Натисніть '3' для виходу: "))  if num== 1:  business()  if num == 2:  economy()  if num==3:  print("Наступний виліт через 3 години")  place = 12 printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бала |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Рівень С**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Математичні вирази часто записуються в інфіксній формі, коли оператори знаходяться між операндами, на які вони діють. Проте також можна використовувати запис у постфіксній формі, в якій оператор з’являється після своїх операндів. Наприклад, інфіксний вираз 3 + 4 записується як 3 4 + у постфіксній формі. Конвертувати інфіксний вираз у постфіксну форму можна за наступним алгоритмом:  Розгляньте питання токенізації математичних виразів. Розбивши введений вираз на токени, програма має трансформувати його з інфіксної форми у постфіксну згідно із заданим алгоритмом.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  def precedence(operator):  if operator in ["-","+"]:  return 1  if operator in ["\*","/"]:  return 2  if operator == "^":  return 3  else:  return -1 operators = [] postfixs =[] expression = input("Інфіксований вираз: ") tokens = [i for i in expression.replace(" ","")] for i in range(len(tokens)-1):  try :  while True:  if tokens[i].isdigit() and tokens[i+1].isdigit():  tokens[i] = tokens[i] + tokens[i+1]  tokens.remove(tokens[i+1])  else:  break  except IndexError:  pass for token in tokens:  if token.isdigit():  postfixs.append(token)  if token in "\*/+-^":  while operators != [] and operators[-1] != "(" and not(precedence(token) > precedence(operators[-1])):  postfixs.append(operators[-1])  operators.remove(operators[-1])  operators.append(token)  if token == "(":  operators.append(token)  if token == ")":  while operators[-1] != "(":  postfixs.append(operators[-1])  operators.remove(operators[-1])  operators.remove("(") while operators != []:  postfixs.append(operators[-1])  operators.remove(operators[-1]) print(" ".join(postfixs)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 6 балів |
| 2. | Отримати значення з постфіксного виразу простіше, оскільки він не містить дужок та має прості правила роботи з операторами. Постфіксний вираз можна обчислити за допомогою алгоритму:  **Створити новий порожній список, *values***  **For кожного токена в постфіксному виразі**  **If токен є числом then**  **Конвертувати його в ціле число та додати в кінець**  ***values***  **Else**  **Видалити елемент з кінця values and call it right**  **Remove an item from the end of values and call it left**  **Apply the operator to left and right**  **Дописати результат у кінець *values***  **Return the first item in values as the value of the expression**  Напишіть програму, яка зчитує математичний вираз в постфіксній формі та виводить його результат.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) def is\_number(s):  try:  float(s)  return True  except ValueError:  return False values = [] expression = input("Постфіксований вираз: ") tokens = [i for i in expression] for i in range(len(tokens)-1):  try :  if tokens[i] == " ":  tokens.remove(" ")  while True:  if tokens[i].isdigit() and tokens[i+1].isdigit():  tokens[i] = tokens[i] + tokens[i+1]  tokens.remove(tokens[i+1])  if tokens[i] in "+-" and tokens[i+1].isdigit():  tokens[i] = tokens[i] + tokens[i + 1]  tokens.remove(tokens[i + 1])  else:  break  except IndexError:  pass for token in tokens:  if is\_number(token):  values.append(int(token))  else:  right = values[-1]  values.remove(values[-1])  left = values[-1]  values.remove(values[-1])  result = eval("{} {} {}".format(left, token , right))  values.append(result) print(values[0]) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 4 бали |
| 3. | Лінійна регресія дозволяє побудувати пряму, яка найкраще наближає набір з n точок даних (data points). Це дає можливість, наприклад, встановити тренд цін на біржі в деякому проміжку часу. Кожна точка з набору має координати (x; y) – хвилину з початку роботи біржі та ціну за акцію. Символами та позначаємо середнє значення по окремим координатам. Лінія тренду представляється рівнянням , де m і b обчислюються за формулами  Напишіть програму, яка зчитує колекцію точок від користувача покоординатно, поки не буде введено порожній рядок. У результаті потрібно вивести формулу для прямої, наприклад, y = 0.95x + 0.1 для координат (1, 1), (2, 2.1) та (3, 2.9).  **КОД:**  import re, datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) coordinates = [] while True:  coord = input("Координати: ")  pattern = "[0-9]+\.?[0-9]\*,[0-9]+\.?[0-9]\*"  if re.match(pattern, coord):  coordinates.append((float(coord[:coord.index(",")]),float(coord[coord.index(",")+1:])))  elif coord == "":  break  else:  print("Неправильні кординати!") x = []; y = [] x2 = []; xy = [] n = len(coordinates) for item in coordinates:  x.append(item[0])  y.append(item[1])  xy.append(item[0]\*item[1])  x2.append(item[0]\*\*2) m = (sum(xy) - (sum(x)\*sum(y)) / n) / (sum(x2) - (sum(x)\*\*2) / n) b = sum(y)/n - m\*sum(x)/n print("y = {:.2f}x + {:.1f}".format(m, b)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 4 бали |
| 4. | Виграшна картка лото містить лінію з 5 чисел, які до цього були витягнуті з мішка. Гравці позначають числа, які були витягнуті, закриваючи їх. У нашому додатку будемо заміняти число на 0 у лото-словнику, якщо воно було оголошено.  Напишіть програму, яка працює зі словниками, що представляють картки лото. Якщо картка містить рядок з 5 нулів, вона вважається виграшною. Програма має формувати 2 картки лото (9 стовпців чисел, кожен з яких відповідає десятку числа, 3 рядка, по 5 чисел у кожному), генерувати випадкове число та заново виводити картку після цього. Як тільки якась із карток виграє, програма завершує роботу з повідомленням про виграш відповідного гравця.  http://s001.radikal.ru/i193/1107/87/916146c47dd1.jpg  Підказка. Оскільки всі числа додатні, знаходження 5 нулів на картці відповідає задачі рівності суми 5 входжень нулю. Можливо, Вам так буде простіше моделювати гру. | 6 балів |
| 5. | Напишіть гру «Однорукий бандит» для консолі. У процесі будуть відображатись три випадкових символи з переліку:  Cherry - u"\U0001F352",  Bell - u"\U0001F514",  Lemon - u"\U0001F34B",  Orange - u"\U0001F34A",  Star - u"\u2606",  Skull - u"\U0001F480"  Гравцю видається кредит у розмірі 100 грн., а кожний запуск коштує 5 грн. Якщо «однорукий бандит» показує два однакових символи, користувач виграє 10 грн, а якщо три – 25 грн. У випадку випадіння трьох дзвіночків (bells) сума виграшу складає 100 грн. Якщо випадає два черепи – гравець додатково втрачає 5 грн, а при отриманні трьох черепів вся сума згорає. Гравець може зняти вигране до початку нової спроби або продовжувати грати, поки на рахунку будуть гроші. | 4 бали |

**Зведена таблиця дня 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Рівень А** | **Бали** | **Рівень В** | **Бали** | **Рівень С** | **Бали** |  |
| **1А1** | **1** | **1В1** | **2** | **1С1** | **6** |  |
| **1А2** | **1** | **1В2** | **3** | **1С2** | **4** |  |
| **1А3** | **1** | **1В3** | **3** | **1С3** | **4** |  |
| **1А4** | **1,5** | **1В4** | **2** | **1С4** | **6** |  |
| **1А5** | **1** | **1В5** | **3** | **1С5** | **4** |  |
| **1А6** | **1** | **1В6** | **2,5** |  |  |  |
| **1А7** | **1** | **1В7** | **3** |  |  |  |
| **1А8** | **1** | **1В8** | **2** |  |  |  |
| **1А9** | **1,5** | **1В9** | **3** |  |  |  |
| **1А10** | **1** | **1В10** | **2** |  |  |  |
| **1А11** | **1,5** | **1В11** | **1,5** |  |  |  |
|  |  | **1В12** | **2** |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  | **Кількість** |
|  |  |  |  |  |  | **28** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  | **Сума** |
|  |  |  |  |  |  | **65,5** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Сума А** | **12,5** | **Сума В** | **29** | **Сума С** | **24** |  |

**День 03. Повторне використання коду та робота з даними**

**Рівень А**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Опишіть рекурсивну функцію GCD, яка повертає найбільший спільний дільник чисел x та y. Обчислення НСД відбувається рекурсивним чином відповідно до формули  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  x = input("Число 1: ")  y = input("Чисол 2: ")  if x.isdigit() and y.isdigit():  x = int(x)  y = int(y)  break  else:  print("Введено НЕ число") def ggd(x,y):  if x == y:  return x  elif x > y:  return ggd(x - y, x)  else:  return ggd(x, y-x) print(ggd(x,y)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 2. | Вивести n-те число Каталана за допомогою рекурсивного алгоритму  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  n = input("Введіть бажане число Каталана: ")  if n.isdigit():  n = int(n)  break  else:  print("Це не число!") def Catalan\_namber(n):  if n >= 2:  c = ((2 \* ((2\*n) - 1)) / (n+1)) \* Catalan\_namber(n-1)  return int(c)  else:  return 1 print(Catalan\_namber(n)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 3. | Опишіть рекурсивну функцію для піднесення числа до цілої додатної степені  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) def is\_number(s):  try:  float(s)  return True  except ValueError:  return False while True:  x = input("Введіть число: ")  n = input("Введіть степінь: ")  if n.isdigit() and is\_number(x):  x = int(x)  n = int(n)  break  else:  print("Введено не додатнє число!") def stepin(x, n):  if n == 0:  return 1  else:  return x \* stepin(x,n-1) print(stepin(x,n)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 4. | Більш ефективне рекурсивне визначення , що охоплює випадок  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) def is\_number(s):  try:  float(s)  return True  except ValueError:  return False while True:  x = input("Введіть число: ")  n = input("Введіть степінь: ")  if is\_number(n) and is\_number(x):  x = int(x)  n = int(n)  break  else:  print("Введено не додатнє число!") print(x,n) def stepin(x, n):  if n == 0:  return 1  elif n > 0 and n % 2 == 0:  return stepin(x, n/2 )\*\*2  elif n > 0 and n % 2 != 0:  return x \* stepin(x,n-1)  elif n < 0 :  return 1 / stepin(x, -n) print(stepin(x,n)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 5. | http://paulbourke.net/fractals/gasket/gasket1.gif(Пил Кантора) Множину Кантора отримують шляхом поділу відрізку на 3 частини та видалення середньої з них. Напишіть рекурсивну функцію, що дозволятиме відобразити аналогічне зображення в консольному додатку. Для візуалізації -того рівня множини Кантора використовують правило  n = 0: на початку для [0, 1];  n > 0: застосувати правило n разів | 1,5 бали |
| 6. | **Градова послідовність (Hailstone Sequence)**. Градова послідовність (градові числа, послідовність Колатца) називається так тому, що її значення зазвичай зростають та зменшуються подібно до форми градових хмар.  ÐÐ°ÑÑÐ¸Ð½ÐºÐ¸ Ð¿Ð¾ Ð·Ð°Ð¿ÑÐ¾ÑÑ hailstones in a cloud  Створіть програму, яка *рекурсивно* генеруватиме градову послідовність чисел, починаючи з деякого введеного цілого числа n. Правила формування послідовності:  Якщо n = 1, послідовність закінчується.  Якщо n – парне, наступне число послідовності буде n/2  Якщо n – непарне, наступне число послідовності = (3 \* n) + 1  Використайте програму, щоб показати, що для числа 27 послідовність складатиметься з 112 елементів, починаючись з 27, 82, 41, 124 та закінчуючись числами 8, 4, 2, 1  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  num = input("Введіть число для градової послідавності: ")  if num.isdigit():  num = int(num)  break  else:  print("Це не додатнє число!") def Hansone\_seq(n):  if n == 1:  return "КІНЕЦЬ!"  else:  if num == n:  print("ПОЧАТОК:"+"\n"+str(n))  if n % 2 == 0:  print(n / 2)  return Hansone\_seq(n / 2)  else:  print(3\* n + 1)  return Hansone\_seq(3\* n + 1) print(Hansone\_seq(num)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |
| 7. | Напишіть програму, яка повністю зчитує файл та дописує в нього ПІБ її автора  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) with open("A7.txt","a+") as file:  file.seek(0)  print("До:\n"+file.read())  file.write("\n"+"Дудник Віталій Романович")  file.seek(0)  print("Після:\n"+file.read()) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 8. | Напишіть програму, яка видалить з файлу всі переходи на новий рядок  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  with open("A8.txt", "r") as file:  text = file.read()  print("До:\n" +text) with open("A8.txt", "w") as file:  text = text.replace("\n"," ")  file.write(text) with open("A8.txt", "r") as file:  print("Після:\n" + file.read()) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 9. | Напишіть програму, яка відкриватиме текстовий документ лише тоді, коли буде введено правильний пароль. Користувач обирає username та password, а потім вони перевіряються перед відкриттям. Створіть випадковий пароль з принаймні 8 символів, він повинен містити принаймні одну маленьку та велику літери, а також спеціальні символи.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) parol = "Abc123#@" user = "Vitaliy" while True:  username = input("Ведіть логін: ")  passsword = input("Ведіть пароль: ")  if parol == passsword and user == username:  with open("A9.txt", "r") as file:  print(file.read())  break  else:  print("Неправильний пароль або логін!") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** |  |
| 10. | Напишіть програму, яка додає номери рядків у файл. Назва вхідного файлу має зчитуватись від користувача з клавіатури, як і назва нового файлу, який буде створювати Ваша програма. Кожний рядок у вихідному файлі має починатися з номеру рядка, за яким іде крапка, пробіл і власне рядок із вхідного файлу.  **КОД:**  import datetime, os def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  infile = input("Введіть назву вхідного файлу: ")  if os.path.exists("{}.txt".format(infile)):  break outfile = input("Введіть назву вихідного файлу: ") with open("{}.txt".format(infile), "r") as file:  text = file.read().split("\n")  print(text)  for item in text:  if item.isspace() or item == "":   text.remove(item)  else:  text[text.index(item)] = str(text.index(item)+1) + ". " + item  print(text) with open("{}.txt".format(outfile),"w") as file:  file.write("\n".join(text)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1 бал |
| 11. | Напишіть програму, яка визначає найдовше слово у файлі. Вона має виводити відповідне повідомлення, яке включає довжину найдовшого слова разом з усіма словами цієї довжини, що зустрічаються в файлі. Вважайте кожну групу непробільних символів словом, навіть якщо вона включає числа чи знаки пунктуації.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) with open("A11.txt","r") as file:  text = file.read().replace("\n", " ").split(" ")  maks = 0  listok = []  for item in text:  if len(item) > maks:  maks = len(item)  listok.clear()  listok.append(item)  elif len(item) == maks:  listok.append(item) print("Довжина найдовшого слова:", maks) print("Слова:", ", ".join(listok)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |
| 12. | Напишіть програму, яка відображає слова, що найчастіше зустрічаються в файлі. Програма має починати роботу зі зчитування назви файлу від користувача. Далі вона повинна знаходити слово, розбиваючи кожен рядок у файлі через пробіли. Усі пунктуаційні знаки мають вилучатись зі слова, а регістр букв ігноруватись.  **КОД:**  import datetime,os from collections import Counter def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  infile = input("Введіть назву вхідного файлу: ")  if os.path.exists("{}.txt".format(infile)):  break  with open("{}.txt".format(infile), "r") as file:  text = file.read().lower()  punctuation = [".", ",", "!", "?", "«","»"]  for i in punctuation:  text = text.replace(i, "")  text = text.split(" ")  count = Counter(text)  for i in count.most\_common(3):  print(i[0], " - ",i[1]) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |
| 13. | Створіть програму, яка підраховує суму всіх введених користувачем чисел, пропускаючи некоректно введені (нечислові) дані. Програма має виводити поточну суму після кожного введеного числа. При вводі нечислових даних виводиться повідомлення про помилку, проте робота програми продовжується. Вихід з програми здійснюється після вводу порожнього рядка. Переконайтесь, що програма працює коректно як для цілих, так і дробових значень.  Підказка: вправа вимагає використання виключень без використання файлів.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) suma = 0 while True:  num = input("Число:")  try:  float(num)  except ValueError:  try:  eval(num)  except SyntaxError:  print("Введено не число!")  except NameError:  print("Введено не число!")  else:  suma += eval(num)  print(suma)  else:  suma += float(num)  print(suma)  if num == "":  break print(suma) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 1,5 бали |
| 14. | Генерування паролю шляхом вибору випадкових символів дає відносно безпечний, проте загалом складний для запам’ятовування пароль. Альтернативним способом є конструювання паролю з двох агнлійських слів з подальшою конкатенацією. Такий пароль простіше запам’ятати, хоча і зламати також.  Напишіть програму, яка зчитує файл зі списком слів, обирає два випадковим чином та з’єднує їх в одне. При створенні паролю переконайтесь, що загальна його довжина знаходиться в діапазоні від 8 до 10 символів, а кожне використане слово складається принаймні з трьох букв. Зробіть так, щоб у паролі кожне слово починалось з великої літери та виведіть пароль для користувача. | 1,5 бали |
| 15. | Операційні системи, побудовані на базі Unix, зазвичай включають утиліту head, яка відображає перші 10 рядків файлу, чия назва передається як аргумент командного рядка. Напишіть програму, яка буде забезпечувати аналогічну поведінку. Виводьте відповідне повідомлення про помилку, якщо вибраний користувачем файл не існує або параметр командного рядка пропущено. | 1,5 бали |
| 16. | Операційні системи, побудовані на базі Unix, зазвичай включають утиліту tail, яка відображає останні 10 рядків файлу, чия назва передається як аргумент командного рядка. Напишіть програму, яка буде забезпечувати аналогічну поведінку. Виводьте відповідне повідомлення про помилку, якщо вибраний користувачем файл не існує або параметр командного рядка пропущено.  Для вирішення цієї задачі може бути використано кілька підходів.   1. Завантажити весь вміст файлу в список, а потім відобразити останні 10 елементів. 2. Зчитувати вміст файлу двічі. Один раз, щоб порахувати рядки, другий – для того, щоб вивести останні 10 з них.   Проте обидва ці рішення не підходять для роботи з великими файлами.  За додаткові 0,5 бала придумайте та реалізуйте підхід, як зробити потрібні дії за одне зчитування файлу, зберігаючи 10 рядків за раз. | 1 бал |
| 17. | Операційні системи, побудовані на базі Unix, зазвичай включають утиліту cat, яка виконує конкатенацію кількох файлів, чиї назви передаються як аргументи командного рядка, та вивести результат. Файли відображаються в порядку їх появи в командному рядку.  Напишіть програму, яка буде виконувати ці задачі. Виводьте відповідне повідомлення про помилку, якщо параметрів командного рядка не було введено. | 1,5 бали |

|  |  |  |
| --- | --- | --- |
| 18. | Однією з найпростіших технік зламування шифрів є частотний аналіз. Він досліджує зашифрований текст, щоб визначити, які символи зустрічаються найчастіше. Потім намагаються відобразити найбільш поширені букви алфавіту на символи зашифрованого повідомлення, що зустрічаються частіше за все. Наприклад, для англійського алфавіту такими буквами є E і T.  Напишіть програму, яка ініціює цей процес, визначаючи та виводячи частоти зустрічності всіх літер у файлі. Ігноруйте пробіли, знаки пунктуації та числа. Програма має бути нечутливою до регістру (case insensitive), розглядаючи символи a і A як еквівалентні. Користувач надаватиме назву файлу через параметр командного рядка. Забезпечте появу повідомлення про помилку, якщо користувач вводить неправильну кількість параметрів або якщо програма не може відкрити файл, заданий користувачем. | 2 бали |
| 19. | Алгоритм шифрування простої перестановки без ключа передбачає записування повідомлення в таблицю по стовпцям. Після того, як текст записано стовпцями, він зчитується порядково. Для використання цього шифру отримувачу та відправника потрібно домовитись про спільний ключ – розмір таблиці.  Наприклад, зашифруємо фразу «**ВРАГ БУДЕТ РАЗБИТ**» (пробіли не викорстовуються)    Для візуальної зручності розіб’ємо загальний текст та зашифрований варіант на блоки    Фактично, для розшифровки потрібно знати лише кількість стовпців у таблиці – це і буде ключ даної криптосистеми.  Напишіть програму, яка шифруватиме та дешифруватиме текст, введений користувачем з ключем, що також отримано від користувача.  Зрозуміло, що такий алгоритм досить просто зламується шляхом підбору ключа з перевіркою на отримання зв’язного тексту | 2 бали |
| 20. | Більш надійною, ніж проста перестановка без ключа (задача 2) є одиночна перестановка по ключу. Тепер у якості ключа будемо використовувати слово, у прикладі – «памир». Таблиця для прикладу виглядає приблизно так:    У перших 2 рядках записано букву та її позицію в слові, отриманому після сортування ключа за алфавітом (памир → аимпр)    Після цього треба переставити стовпці в природному порядку:    Зчитавши порядково таблицю, отримаємо зашифрований текст:  ГРДВББФЕРИУЗТТА.  Щоб розшифрувати текст, досить знати ключове слово: його довжина визначить кількість стовпців, а букви – порядок перестановки.  Модифікуйте програму із задачі 2 для використання цього алгоритму | 2 бали |
| 21. | Скитала — пристрій шифрування, який використовувався у Стародавній Спарті. Вона являла собою дерев'яний циліндр, на який намотувалась шкіряна стрічка. Перпендикулярно стрічці писалось повідомлення, потім стрічка розмотувалась і передавалась одержувачу.  Повідомлення НАСТУПАЙТЕ при його розміщенні по окружності стержня по 3 букви дає шифртекст НУТАПЕСА\_ТЙ  http://crypto-r.narod.ru/glava2/Image30.gif  https://upload.wikimedia.org/wikipedia/commons/5/51/Skytale.png  Окружність стержня і виступатиме ключем для шифрування та розшифровки. Напишіть програму, яка буде виконувати дані дії. | 1,5 бали |
| 22. | Щоб зашифрувати повідомлення за допомогою шифру частоколу, переписуємо його у відповідному вигляді. Наприклад, для слова «криптографія» з висотою частоколу 2 отримаємо таку схему: кРиПтОгРаФіЯ. Далі зчитуємо текст рядками, почавши з верхнього. В результаті отримуємо криптотекст «рпорфякитгаі». "Висоту" частоколу (секретний ключ шифру) можна вибирати, що й буде запропоновану користувачу.  Напишіть програму, що буде зашифровувати та розшифровувати введений користувачем текст за допомогою введеного ключа – висоти частоколу. | 1,5 бали |
| 23. | Простий алгоритм шифрування даних пропонує записувати алфавіт в одному рядку, а в іншому – ті ж букви, але в іншому порядку. Другий рядок називається криптографічним ключем.    Кожна з букв початкового тексту заміняється на відповідну букву ключа.  Напишіть програму, яка буде зчитувати рядок до 100 символів та замінятиме маленькі букви вхідного рядка на символи ключа. Потім програма повинна дешифрувати отриманий рядок та відновити початкове повідомлення. | 1,5 бали |

**Рівень В**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | Трикутник Паскаля – це трикутник чисел, який містить біноміальні коефіцієнти, що знаходяться за формулою  Напишіть функцію, яка будуватиме трикутник Паскаля заданої висоти (аргумент функції). Наприклад,  1  1 1  1 2 1  1 3 3 1  1 4 6 4 1  Зверніть увагу на форматування виводу трикутника.  **КОД:**  from math import factorial import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) a = lambda n,r: int(factorial(n) / (factorial(r) \* factorial(n-r))) while True:  height = input("Введіть висоту трикутника: ")  if height.isdigit():  break  else:  print("Це не число") height = int(height) for n in range(height):  for r in range(n+1):  if r == 0:  print(" "\*(height-n)+str(a(n,r)),end=" ")  else:  print(a(n,r),end=" ")  print("") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2,5 бали |
| 2. | Редакційна відстань (edit distance) між двома рядками є мірою їх схожості. Чим менша редакційна відстань, тим більше схожі між собою рядки відносно мінімальної кількості вставок, видалень та замін при перетворенні одного рядка в інший.  Наприклад, рядки kitten та sitting. Перший рядок можна перетворити в другий, якщо: замінити k на s, замінити e на i, вставити g в кінець рядка. Три - це найменша кількість операцій, що можна виконати для перетворення kitten у sitting.  Напишіть рекурсивну функцію, яка обчислює редакційну відстань між 2 рядками. Використовуйте наступний алгоритм:  Візьмемо рядки s і t  If довжина s = 0  Повернути довжину t  Else if довжина t = 0  Повернути довжину s  Else  cost = 0  If останній символ в s не дорівнює останньому символу в t  cost = 1  d1 = (редакційна відстань між всіма символами, крім  останнього в s та всіма символами в t) + 1  d2 = (редакційна відстань між всіма символами в s та всіма  символами в t, крім останнього) + 1  d3 = (редакційна відстань між всіма символами, крім  останнього, в s та всіма  символами в t, крім останнього) + 1  Return min(d1, d2, d3)  Використайте дану рекурсивну функцію, щоб написати програму, яка зчитує 2 рядки від користувача і виводить редакційну відстань між ними.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) def distance(s, t):  if len(s) == 0:  return len(t)  elif len(t) == 0:  return len(s)  else:  if s[-1] == t[-1]:  return distance(s[:-1], t[:-1])  d1 = distance( s[:-1], t)+1  d2 = distance( s, t[:-1])+1  d3 = distance( s[:-1], t[:-1] )+1  return min(d1, d2, d3) s = input("1: ") t = input("2: ") print(distance(s, t)) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 3 бали |
| 3. | У романі “Gadsby” Ернеста Вінсента Райта понад 50,000 слів. Основною особливістю оригіналу є те, що в жодному з цих слів не зустрічається буква “e” – найбільш поширена буква в англійському алфавіті.  Напишіть програму, яка зчитує список англійських слів з файлу та визначає, яка частина слів використовує кожну букву англійського алфавіту. Відобразіть результат для усіх 26 літер. Доповніть вивід повідомленням щодо того, яка буква найрідше зустрічається у файлі. Програма має ігнорувати пунктуацію та не розрізняти великі та малі букви.  **КОД:**  import re import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) points =[] file = open("3B3.txt") text = file.read().lower() print(text) for i in "abcdefghijklmnopqrstuvwxyz":  print("The letter {} is found {} times".format(i.upper(),len(re.findall(i, text)))) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |
| 4. | Python використовує символ #, щоб позначати початок коментарів. Коментар закнічується в кінці рядка, що містить символ #.  Напишіть програму, яка видалятиме всі коментарі з первинного коду, написаного мовою Python. Перевіряйте кожен рядок файлу, щоб визначити, чи присутній в ньому символ #. Якщо це так, програма має видалити всі символи від # до кінця рядка (ситуацію, коли коментар знаходиться всередині рядка, проігноруємо). Збережіть змінений файл, використовуючи нову назву, що буде вводитись користувачем. Також користувач має вводити назву вхідного файлу.  Переконайтесь, що з’являється відповідне повідомлення про помилку, якщо при доступі до файлу виникла проблема.  **КОД:**  import re, datetime, os def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) while True:  infile = input("Назва вхідного файла: ")  if os.path.exists("%s.py"%infile):  break outfile= input("Введіть нову назву: ") new\_text = [] with open("%s.py"%infile, "r+") as file:  text = file.read().split("\n")  for row in text:  amount = re.findall("#", row)  if len(amount) >= 1:  new\_text.append(row[:row.index("#")])  else:  new\_text.append(row)  for i in new\_text:  if i.isspace():  new\_text.remove(i)  file.truncate(0)  file.write("\n".join(new\_text)) os.rename("%s.py"%infile, "%s.py"%outfile) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |

|  |  |  |
| --- | --- | --- |
| 5. | Секретна інформація часто редагується чи видаляється з документів перед їх публікуванням. Після публікації документів відредагований текст часто заміняється чорними прямокутниками.  Напишіть програму, яка редагує всі секретні слова, заміняючи їх зірочками, навіть якщо слово з’являється всередині іншого слова. Список секретних слів постачається в окремому файлі. Збережіть відредаговану версію початкового файлу в новий файл.  Корисним може бути метод replace() для роботи з рядками. Доповніть програму незалежністю від регістру літер у тексті, тобто слова exam, Exam, ExaM та EXAM будуть вважатись однаковими. | 2,5 бали |
| 6. | Run-length encoding (RLE) – простий підхід до стиснення даних, який може бути ефективним, коли на сусідніх позиціях у списку присутні повторювані значення. Стиснення досягається за рахунок заміни групи повторюваних значень на одну його копію, після якої йде кількість повторів цього значення. Наприклад,  ["A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "B", "B", "B", "B", "A", "A", "A", "A", "A", "A", "B"]  буде стиснуто до ["A", 12, "B", 4, "A", 6, "B", 1]. Розпаковка виконується шляхом виведення значення зі списку стільки разів, скільки вказано.  Напишіть програму, яка зчитуватиме файл з текстом, записуватиме його стиснений варіант у compressed.txt, а розпакований варіант – у decompressed.txt. Також програма має визначати ступінь стиснення тексту у відсотках.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) file = open("3B6.txt") text = file.read() print(text) symbol= [] colich=0 start=0 finish = 1 while finish != len(text):  symbol.append(text[start])  while len(set(text[start:finish]))==1:  colich+=1  finish+=1  symbol.append(colich)  colich=0  start= finish-1 print("Текст був зжатий на {} %".format(100-(100\*len(symbol)/len(text)))) with open("compressed.txt","w")as file\_comp:  for i in symbol:  file\_comp.write(str(i)) with open("decompressed.txt","w")as file\_decomp:  for i in symbol:  if symbol.index(i)%2==0:  t= i\*(symbol[symbol.index(i)+1])  file\_decomp.write(t) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 3 бали |
| 7. | Один з перших відомих прикладів шифрування використовував Юлій Цезар. Йому потрібно було надсилати письмові інструкції своїм генералам, проте він не хотів, щоб вороги дізнались про його плани, якщо повідомлення потрапить їм у руки. Тому Цезар запропонував просту ідею: кожна літера первинного повідомлення зсувалась на 3 позиції в алфавіті. Замість A записували D, замість B – E, C ставало F, D – G і т. д. Останні 3 букви в алфавіті мали замінники з його початку (X = A, Y = B, Z = C). Небуквенні символи не зашифровувались.  Напишіть програму, яка реалізує шифр Цезаря. Дозвольте користувачеві ввести повідомлення та розмір зсуву, а потім виведіть «зсунуте» повідомлення. Переконайтесь, що Ваша програма коду як великі, так і маленькі букви. Додаток також має підтримувати від’ємний зсув, щоб мати можливість як зашифрувати, так і розшифрувати повідомлення.  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) k = int(input("Key: ")) plaintext = input("plaintext: ") print("ciphertext: ", end="") for i in plaintext:  if 64 < ord(i) < 91:  ciphertext = chr(((((ord(i) + k) - 65) % 26) + 65))  print(ciphertext, end="")  elif 96 < ord(i) < 123:  ciphertext = chr(((((ord(i) + k) - 97) % 26) + 97))  print(ciphertext, end="")  else:  print(i, end="") print("\n") printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2 бали |
| 8. | Код Морзе – схема кодування, яка використовує точки та тире для представлення цифр і букв. Напишіть програму, яка буде використовувати словник для зберігання відображення букв і цифр на код Морзе (таблиця).    Ваша програма має читати повідомлення користувача та перетворювати його в код Морзе з наступним виводом закодованого тексту. Пробіли між словами слід залишати, а решту символів, що не є цифрою або буквою – ігнорувати. Наприклад, Hello, World! буде мати код  **.... . .-.. .-.. --- .-- --- .-. .-.. -..**  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) dictionary = {"A" : ".-", "B" : "-...", "C" : "-.-." , "D" : "-..", "E" : ".",  "F" : "..-.", "G" : "--.", "H" : "....", "I" : "..",  "J" : ".---", "K" : "-.-", "L" : ".-..", "M" : "--",  "N" : "-.", "O" : "---", "P" : ".--.", "Q" : "--.-",  "R" : ".-.", "S" : "...", "T" : "-", "U" : "..-",  "V" : "...-", "W" : ".--", "X" : "-..-", "Y" : "-.--",  "Z" : "--..", "0" : "-----","1" : ".----", "2" : "..---",  "3" : "...--","4" : "....-","5" : ".....", "6" : "-....",  "7" : "--...","8" : "---..","9" : "----."} text = input("Введіть текст: ").upper() for letter in text:  if letter not in dictionary:  text = text.replace(letter,"") for key, value in dictionary.items():  text = text.replace(key,value) print(text) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 3 бали |

|  |  |  |
| --- | --- | --- |
| 9. | Поширена ширина вікна терміналу складає 80 символів, проте існують як вужчі, так і ширші. Це може призвести до проблем із відображенням частин тексту, оскільки рядки можуть бути надто довгі або, навпаки, займати мало місця, утворюючи багато порожніх частин.  Напишіть програму, яка відкриває файл та виводить його вміст настільки заповнюючи екран, наскільки це можливо. Якщо зчитаний рядок надто довгий, програма має розбити його на слова і додавати слова до поточного рядка, поки він не буде повним. Далі програма має починати новий рядок та виводити решту слів. Аналогічно, якщо зчитаний рядок надто короткий, потрібно буде використовувати слова з наступного рядка для закінчення заповнення. Наприклад:    При форматуванні з довжиною рядка 50 символів цей текст буде виводитись так:    Забезпечте коректну роботу програми для тексту з багатьма абзацами.  Підказка: використовуйте константу для представлення максимальної довжини рядка. Це спрощує оновлення програми для зміни розміру вікна. | 2,5 бали |

|  |  |  |
| --- | --- | --- |
| 10. | Створіть Python-програму, яка зчитує кілька файлів з первинним кодом Python та ідентифікує функції, тіло яких починається не з коментаря. Для цієї вправи вважаємо, що будь-який рядок, що починається з def та пробілу є початком визначення функції. Припускається, що символ коментування, #, буде першим символом, який знаходиться на попередньому рядку відносно оголошення функції. Виведіть назви всіх функцій, у яких немає коментарів, разом з назвою файлу та номером рядка, в якому відбувається оголошення цієї функції. | 3 бали |
| 11. | Перевірка правопису допомагає людям, які не можуть записувати їх коректно. Напишіть програму, яка зчитуватиме файл та відображатиме всі слова, які неправильно написані. Визначення некоректно написаних слів відбувається шляхом порівняння зі списком уже відомих слів. Усі слова, які не зустрічаються в такому «словнику» вважаються помилковими. Ігноруйте регістр букв у словах та знаки пунктуації при перевірці.  Підказка: пошук у списку за допомогою оператора in досить повільний. Краще працювати зі словником та перевіряти його наявність серед ключів або значень. Тоді слова стануть ключами, а значення можуть бути, наприклад, нулями (не важливо, оскільки вони не використовуються).  **КОД:**  import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) with open("B11(dict).txt") as file:  text = file.read().split(", ")  print(text)  dictionary = {i : 0 for i in text} with open("B11(text).txt") as file:  text = file.read().lower().replace(",","").split(" ")  print(text) print("Непрвильні слова:") for word in text:  if word not in dictionary:  print(word) printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 2,5 бали |
| 12. | Реалізуйте гру «Хрестики-нолики». Крім головної функції, виділіть окремі підпрограми для   * + - ініціалізації матриці 3х3 «порожніми» символами,     - обробки ходу гравця,     - обробки ходу комп’ютера,     - перевірки виграшу,     - виводу результатів.   Нехай комп’ютер під час свого ходу заповнює першу «порожню» клітинку, а гравець – задає координати (X,Y), які треба перевірити на валідність. | 3 бали |

**Рівень С**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
|  |  |  |
| . |  |  |
| 3. | Фрактали є самоподібними структурами, тому природною є рекурсивна структура алгоритму їх побудови. Напишіть програму, яка буде виводити на екран килим Серпінського.  ../_images/sierpinski.png  Ідея побудови килиму наступна. Починаємо з великого залитого одним кольором трикутника. Ділимо його на 4 частини, з’єднавши середини кожної зі сторін трикутника (темносиній трикутник на рисунку). Продовжуємо робити так і для отриманих трикутників (червоні трикутники, потім зелені, білі та жовті). Таким чином, Вам будуть потрібні функції, що   * знаходитимуть середини сторін заданого трикутника (визначається покоординатно) * відрисовуватимуть сам трикутник, наприклад, за допомогою turtle   формуватимуть нові трикутники за вище наведеними правилами  **КОД:**  import turtle import datetime def printTimeStamp(name):  print('\nАвтор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) def drawTriangle(points,color,myTurtle):  myTurtle.fillcolor(color)  myTurtle.up()  myTurtle.goto(points[0][0],points[0][1])  myTurtle.down()  myTurtle.begin\_fill()  myTurtle.goto(points[1][0],points[1][1])  myTurtle.goto(points[2][0],points[2][1])  myTurtle.goto(points[0][0],points[0][1])  myTurtle.end\_fill() def getMid(p1,p2):  return ( (p1[0]+p2[0]) / 2, (p1[1] + p2[1]) / 2) def sierpinski(points,degree,myTurtle):  colormap = ['blue','red','green','white','yellow',  'violet','orange']  drawTriangle(points,colormap[degree],myTurtle)  if degree > 0:  sierpinski([points[0], getMid(points[0], points[1]),  getMid(points[0], points[2])], degree-1, myTurtle)   sierpinski([points[1], getMid(points[0], points[1]),  getMid(points[1], points[2])], degree-1, myTurtle)   sierpinski([points[2], getMid(points[2], points[1]),  getMid(points[0], points[2])], degree-1, myTurtle) def main():  myTurtle = turtle.Turtle()  myWin = turtle.Screen()  myPoints = [[-140,-70],[0,140],[140,-70]]  sierpinski(myPoints,3,myTurtle)  myWin.exitonclick() main() printTimeStamp("Коробов Даніїл, Дудник Віталій")  **ВИВІД:** | 5 балів |
|  |  |  |

**Зведена таблиця дня 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Рівень А** | **Бали** | **Рівень В** | **Бали** | **Рівень С** | **Бали** |  |
| **3А1** | **1** | **3В1** | **2,5** | **3С1** | **5** |  |
| **3А2** | **1** | **3В2** | **3** | **3С2** | **6** |  |
| **3А3** | **1** | **3В3** | **2** | **3С3** | **5** |  |
| **3А4** | **1** | **3В4** | **2** | **3С4** | **5** |  |
| **3А5** | **1,5** | **3В5** | **2,5** |  |  |  |
| **3А6** | **1,5** | **3В6** | **3** |  |  |  |
| **3А7** | **1** | **3В7** | **2** |  |  |  |
| **3А8** | **1** | **3В8** | **3** |  |  |  |
| **3А9** | **1,5** | **3В9** | **2,5** |  |  |  |
| **3А10** | **1** | **3В10** | **3** |  |  |  |
| **3А11** | **1,5** | **3В11** | **2,5** |  |  |  |
| **3А12** | **1,5** | **3В12** | **3** |  |  |  |
| **3А13** | **1,5** |  |  |  |  |  |
| **3А14** | **1,5** |  |  |  |  |  |
| **3А15** | **1,5** |  |  |  |  |  |
| **3А16** | **1** |  |  |  |  |  |
| **3А17** | **1,5** |  |  |  |  |  |
| **3А18** | **1,5** |  |  |  |  |  |
| **3А19** | **1,5** |  |  |  |  |  |
| **3А20** | **1,5** |  |  |  |  | **Кількість** |
| **3А21** | **1,5** |  |  |  |  | **39** |
| **3А22** | **1,5** |  |  |  |  |  |
| **3А23** | **1,5** |  |  |  |  |  |
|  |  |  |  |  |  | **Сума** |
|  |  |  |  |  |  | **82,5** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Сума А** | **30,5** | **Сума В** | **31** | **Сума С** | **21** |  |