МІНІСТЕРСТВО ОСВІТИ І НАУКИ України

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

Циклова комісія програмування

**Звіт**

**з трьох ДНів ПРАКТИКИ**

**з дисципліни «Основи програмування»**

виконала студентка

групи 1П-17

Москаленко В.І.

перевірив викладач

Марченко С.В.

Черкаси, 2019

**Рівень А**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | 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 відношеням  Виведіть відповідне значення ІМТ; вони будуть аналізуватись у наступних задачах.  import datetime  a = str(input("Дюйми або метри (D/M)"))  rost = int(input("Введіть ваш зріст"))  ves = int(input("Введіть вашу вагу"))  if a == 'D':  res = (703 \* (ves / rost \*\* 2)\*10)  print(float('{:.2f}'.format(res)))  elif a == 'M':  res = ((ves / rost \*\* 2)\*10000)  print(float('{:.2f}'.format(res)))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 2. | 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.  import datetime  a = float(input("Введіть тиск: "))  b = (a \* 0.14503773773)  print(float('{:.10f}'.format(b)))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 0,5 бала |
| 3. | 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.  import datetime  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  while True:  buk = input("Введiть букву:")  if buk == "a":  print("Голосна")  elif buk == "e":  print("Голосна")  elif buk == "i":  print("Голосна")  elif buk == "o":  print("Голосна")  elif buk == "u":  print("Голосна")  elif buk == "y":  print("Iнколи голосна,а iнколи - приголосна")  else:  print("Приголосна")  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 4. | 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.  import datetime  name = str(input("Введить назву страви: "))  kilk = int(input("Кількість страв яку ви замовляли: "))  x = float(input("яка ціна за їжу: "))  b = float(kilk\*x)  chay = float(b\*0.14)  pod = float(b\*0.18)  zag = float(chay + b + pod)  print("Розмір податку: " + '{:.2f}'.format(pod), "\n", "Чайові: " + '{:.2f}'.format(chay), "\n", "Загальна сумма: " '{:.2f}'.format(zag))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 5. | Напишіть програму, яка спочатку зчитує температуру в градусах Цельсія. Далі вона має вивести еквіваленту температуру в градусах Кельвіна та Фаренгейта.  import datetime  c = float(input("Температура в градусах Цельсiя:"))  k = c + 273.15  f = (c \* 9/5) + 32  print("Градуси Кельвiна:", k, "\n" "Градуси Фаренгейта:", f)  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 0,5 бала |
| 6. | 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).  import datetime  a = int(input("Введіть 4-значне число: "))  s = str(a)  a = int(s[0])  b = int(s[1])  c = int(s[2])  d = int(s[3])  print(a+b+c+d)  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 7. | Поширеною думкою є те, що 1 рік людського життя еквівалентний 7 рокам життя собами. 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.  import datetime  chel = int(input("Людськi роки:"))  if 0 < chel <= 2:  pes = 10.5 / chel  print("Собачi роки:", pes)  elif chel > 2:  pes = 10.5 + (chel \* 4)  print("Собачi роки:", pes)  elif chel < 0:  print("Введено вiд'ємне число")  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1,5 бали |
| 8. | 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.  import datetime  shtu = int(input("Скільки штучок: "))  shtu2 = int(input("Скільки штукенцій: "))  a = shtu \* 75 + shtu2 \* 112  print("Загальна маса: " + str(a))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 9. | 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.  import datetime  ros = float(input("Який ваш зріст: "))  d = ros / 2.54  fot = d / 12  print("Ріст у дюймах: " + str(d) + "Ріст у футах: " + str(fot))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 10. | Напишіть програму, яка зчитує проміжок часу у вигляді кількості днів, годин, хвилин та секунд. Обчисліть та виведіть загальну кількість секунд у цьому проміжку часу.  import datetime  dni = int(input("Кількість днів: "))  god = int(input("Кількість годин: "))  xv = int(input("Кількість хвилин: "))  sec = int(input("Кількість секунд: "))  a = dni \* 86400 + god \* 3600 + xv \* 60 + sec  print("Всього секунд: " + str(a))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 12. | 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 середньго.  import datetime  a = 0  b = 0  while True:  c = int(input("Введіть значення: "))  if c == 0:  break  else:  a += c  b += 1  if b == 0:  print("Не можно вводити перше число 0")  else:  print("Середнє значення: " + str(a/b))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Литвиненко Чубенко") | 1 бал |
| 13. | 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.  import datetime  start = float(input("Грошей покладено на рахунок:"))  y1 = start + (start \* 0.14)  y2 = y1 + (y1 \* 0.14)  y3 = y2 + (y2 \* 0.14)  print("1 рiк:", float('{:.2f}'.format(y1)))  print("2 рiк:", float('{:.2f}'.format(y2)))  print("3 рiк:", float('{:.2f}'.format(y3)))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 14. | 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.  import datetime  a = int(input("Скільки вчорашных буханок ви хочете купити: "))  b = a \* 8.5  c = (a \* 8.5)\*0.4  d = b - c  print("Звичайна ціна: " + str(b) + " Ваша снижка: " + str(d) + " Загальна ціна: " + str('{:.2f}'.format(c)))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1 бал |
| 15. | Напишіть програму, яка зчитує ціле число n від користувача та показує суму всіх цілих чисел від 1 до n. Формула для знаходження цієї суми:  import datetime  n = int(input("Введiть цiле число:"))  nast = n + 1  mnoj = n \* nast  dil = mnoj / 2  print(dil)  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 0,5 бала |
| 16. | 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.  import datetime  import math  r = int(input("Введiть радiус:"))  krug = math.pi \* (r\*\*2)  kulya = 4/3 \* math.pi \* (r\*\*3)  print("Площа кругу:", float('{:.5f}'.format(krug)), "\n", "Об'єм кулi:", float('{:.5f}'.format(kulya)))  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 0,5 бала |
| 17. | 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.  import datetime  import math  s = int(input("Довжина сторони:"))  n = int(input("Кiлькiсть сторiн:"))  pl = ((n \* (s\*\*2)) / (4 \* math.tan(math.pi / n)))  print("Площа полiгону:", float('{:.5f}'.format(pl))) | 0,5 бала |
| 18. | 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.  a = int(input ("Введіть кількість секунд: "))  god = str(a//3600)  min = (a//60)%60  sec = a%60  if min<10:  min='0'+str(min)  else:  min=str(min)  if sec<10:  sec='0'+str(sec)  else:  sec=str(sec)  print(god +':' + min + ':' + sec) | 1,5 бали |
| 19. | 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.  import datetime  a = str(input("Позначки:"))  if a == "Вiдпустка" or "вiдпустка":  print("Будильник вимкнено")  elif a == "Вихiдний" or "вихiдний":  print("Будильник вимкнено")  else:  print("Будильник ввiмкнено") | 1 бал |
| 21. | **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 ().  import math  import datetime  m = 67  c = 3.7  p = 1.038  K = 5.4 \* (10\*\*3)  Tw = 100  Ty = 70  def temp(name, T0):  t = ((m\*\*2/3) \* c \* (p \*\* 1/3)) / K \* (math.pi\*\*2) \* ((4 \* math.pi)/3)\*\*(2/3) \* math.log1p(0.76 \* ((T0 - Tw)/(Ty - Tw)))  print(name, round(t, 4), "сек")  temp("З холодильника:" ,4)  temp("Кiмнатна температура:" ,20)  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | 1,5 бала |

**Таблиця дня 1 Рівень А**

|  |  |
| --- | --- |
| **Рівень А** | **Бали** |
| **А1** | **1** |
| **А2** | **0,5** |
| **А3** | **1** |
| **А4** | **1** |
| **А5** | **0,5** |
| **А6** | **1** |
| **А7** | **1,5** |
| **А8** | **1** |
| **А9** | **1** |
| **А10** | **1** |
| **А11** | **1** |
| **А12** | **1** |
| **А13** | **1** |
| **А14** | **1** |
| **А15** | **0,5** |
| **А16** | **0,5** |
| **А17** | **0,5** |
| **А18** | **1,5** |
| **А19** | **1** |
| **А20** | **----** |
| **А21** | **1,5** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Сума А** | **18.5** |

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Умова задачі** | | **Бали /**  **Зв’язки** |
| 1. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  p = int(input("Введіть тиск: "))  v = int(input("Введіть об`єм: "))  t = int(input("Введіть температуру: "))  r = 8.314  k = t + 273.15  print("Молярна масса: ", (p\*v/1000)/(r\*k)) | | 2 бали |
| 2. | 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).  import datetime  path = float(input('Введіть магнітуду(ML):'))  if path <= 2.0 :  print('Мікро \nМікроземлетруси, не відчуваються.')  elif path >= 2.0 and path <= 3.0 :  print('Дуже слабкий \nЯк правило не відчуваються, але реєструються.')  elif path >= 3.0 and path <= 4.0 :  print('Слабкий \nЧасто відчуваються, дуже рідко завдають шкоди.')  elif path >= 4.0 and path <= 5.0 :  print('Легкий\nВідчутне тремтіння речей всередині будинків, значна шкода малоймовірна.')  elif path >= 5.0 and path <= 6.0 :  print('Помірний \nМоже завдати значної шкоди старим та погано сконструйованим будівлям на незначній території.')  elif path >= 6.0 and path <= 7.0 :  print('Сильний \nМоже спричинити руйнацію на території до 150 км довжиною/шириною в населених регіонах.')  elif path >= 7.0 and path <= 8.0 :  print('Дуже сильний \nЗначна руйнація на значній території.')  elif path >= 8.0 and path <= 10.0 :  print('Великий \nСерйозна руйнація на територіях довжиною/шириною в сотні кілометрів.')  elif path >= 10.0 :  print('Рідкісно великий') | | 1,5 бали |
| 3. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  xv = int(input("Скільки ви використали хвилин: "))  sms = int(input("Скільки смс ви використали: "))  a = 35  b = (a\*0.05) + 0.44 + a  if xv>200:  a += (xv-200)\*0.17  if sms>50:  a += (sms - 50)\*0.17  print("Рахунок без податків: " + str('{:.2f}'.format(b)))  print("Загальний рахунок: " + str('{:.2f}'.format(b))) | | 2 бали |
| 4. | 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    import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  num1 = int(input("Введіть першу цифру : "))  num2 = int(input("Введіть другу цифру : "))  num3 = int(input("Введіть третю цифру : "))  num4 = int(input("Введіть четверту цифру : "))  num5 = int(input("Введіть пяту цифру : "))  num6 = int(input("Введіть шосту цифру : "))  num7 = int(input("Введіть сьому цифру : "))  num8 = int(input("Введіть восьмуцифру : "))  num9 = int(input("Введіть девяту цифру : "))  num10 = int(input("Введіть десяту цифру : "))  num11 = int(input("Введіть одинадцяту цифру : "))  num12 = int(input("Введіть дванадцяту цифру : "))  nepar\_sum = num1 + num3 + num5 + num7 + num9 + num11  par\_sum = num2 + num4 + num6 + num8 + num10 + num12  par\_sum\_times3 = par\_sum\*3  total\_nepar\_par\_times3=nepar\_sum + par\_sum\_times3  total\_rem10 = total\_nepar\_par\_times3 % 10  if total\_rem10 == 0:  isbn13 = 0  else:  isbn13 = 10 -total\_rem10  print("Перевірочною цифрою є : ", isbn13)  printTimeStamp('Москаленко, Яковенко')  input('\n') | | 2,5 бала |
| 5. | 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.  from math import floor  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Литвиненко Чубенко")  def main():  while True:  a = int(input("Введіть сумму: "))  b = floor(a)  if b > 0:  break  g5 = b // 500  g2 = (b % 500) // 200  g1 = ((b % 500) % 200) // 100  k50 = (((b % 500) % 200) % 100) // 50  k25 = ((((b % 500) % 200) % 100) % 50) // 25  print(g5,g2,g1,k50,k25)  if \_\_name\_\_ == "\_\_main\_\_":  main() | | 2,5 бала |
| 6. | 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 | |  |  |  |   import datetime  print("Місце проживання (Гуртожиток, Квартира, Будинок)")  home = input('Введіть місце проживання: ')  time = int(input('Введіть скільки годин ви приблизно знаходитесь дома: '))  print(' ')  if home == 'Гуртожиток':  if time < 6:  print('Тобі підійде: Мурашник')  elif time == 6:  print('Тобі підійде: Мурашник або рибки')  if time > 6:  print('Тобі підійдуть: Рибки')  if home == 'Квартира':  if time < 10:  print("Тобі підійде: Хом'як")  if home == 'Квартира' and time == 10:  print("Тобі підійде: Хом'як або Кішка")  if home == 'Квартира' and time > 10:  print('Тобі підійде: Кішка')  if home == 'Будинок' and time <= 10:  print("Тобі підійде: Змія")  if home == 'Будинок' and time > 10 and time < 17:  print("Тобі підійде: Собака")  if home == 'Будинок' and time > 18:  print('Тобі підійде: В’єтнамське порося')  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | | 2,5 бала |
| 9. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  result = ''  q = int(input("число: "))  while q > 0:  r = q % 2  result = str(r) + result  q = q // 2  print(result, "\nІніціалізувати змінну result порожнім рядком\nОголосити змінну | | 2,5 бали |
| 10. | 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.  import math  t1 = int(input("Введіть першу широту - "))  t2 = int(input("Введіть другу широту - "))  g1 = int(input("Введіть першу довготу - "))  g2 = int(input("Введіть другу довготу - "))  t1 = math.radians(t1)  t2 = math.radians(t2)  g1 = math.radians(g1)  g2 = math.radians(g2)  print ("Прерша широта в радіанах: ", (t1))  print ("Друга широта в радіанах: ", (t2))  print ("Перша довгота в радіанах: ", (g1))  print ("Друга довгота в радіанах: ", (g2))  V=6371.01 \* math.acos(math.sin(t1) \* math.sin(t2) + math.cos(t1) \* math.cos(t2) \* math.cos(g1 - g2))  print(V)  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name="Москаленко, Яковенко"  printTimeStamp(name) | | 3 бали |
| 11. | 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.  import math  t1 = 46.  t2 = 49  t3 = 48  g1 = 30  g2 = 24  g3 = 22  t1 = math.radians(t1)  t2 = math.radians(t2)  t3 = math.radians(t3)  g1 = math.radians(g1)  g2 = math.radians(g2)  g3 = math.radians(g3)  print ("Перша широта в радіанах: ", (t1))  print ("Друга широта в радіанах: ", (t2))  print ("Третя широта в радіанах: ", (t3))  print ("Перша довгота в радіанах: ", (g1))  print ("Друга довгота в радіанах: ", (g2))  print ("Третя довгота в радіанах: ", (g3))  V1=6371.01 \* math.acos(math.sin(t1) \* math.sin(t2) + math.cos(t1) \* math.cos(t2) \* math.cos(g1 - g2))#Відстань між Одесою та Львовом  print(V1)  V2=6371.01 \* math.acos(math.sin(t1) \* math.sin(t3) + math.cos(t1) \* math.cos(t3) \* math.cos(g1 - g3))#Відстань між Одесою та Харьковим  V3=6371.01 \* math.acos(math.sin(t3) \* math.sin(t2) + math.cos(t3) \* math.cos(t2)\* math.cos(g3 - g2))#Відстань між Харьковом та Львовом  p = (V1+V2+V3)/2  S = math.sqrt(p\*(p-V1)\*(p-V2)\*(p-V3))  print("Площа трикутника - ",S)  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name="Москаленко, Яковенко"  printTimeStamp(name) | | 2 бали  В10 |
| 12. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  gor = {  0: 'Rat',  1: 'Ox',  2: 'Tiger',  3: 'Hare',  4: 'Dragon',  5: 'Snake',  6: 'Horse',  7: 'Sheep',  8: 'Monkey',  9: 'Rooster',  10: 'Dog',  11: 'Pig',  }  print(gor[(int(input('>>> Year = ')) - 2008) % 12])  printTimeStamp('Москаленко, Яковенко')  input('\n') | | 2,5 бали |
|  | |
| 14. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  TGR = 13.12  fac1 = 0.6215  fac2 = -11.37  fac3 = 0.3965  DRG = 0.16  temp = float(input("Введіть температуру повітря(в Цельсіях): "))  speed = float(input("Введіть швидкість вітру(км/год): "))  tfd = TGR + (fac1 \* temp) + (fac2 \* speed \*\* DRG) + (fac3 \* temp \* speed \*\* DRG)  print("Ваш індекс вітру ","%d"%(tfd))  printTimeStamp('Москаленко, Яковенко')  input('\n') | | 1,5 бали |
| 15 | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  year = int(input("Введіть рік: "))  if year % 400 != 0:  print("Звичайний рік")  elif year % 100 == 0:  if year % 4 == 0:  print("Високосний рік")  else:  print("Звичайний рік")  else:  print("Високосний рік") | | 1,5 бали |
| 18. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  a = input('Введiть значення клiтки: ')  if a == 'a1' or a == 'a3' or a == 'a5' or a == 'a7':  print('Клітинка чорна')  elif a == 'a2' or a == 'a4' or a == 'a6' or a == 'a8':  print('Клітинка біла')  elif a == 'b1' or a == 'b3' or a == 'b5' or a == 'b7':  print('Клітинка біла')  elif a == 'b2' or a == 'b4' or a == 'b6' or a == 'b8':  print('Клітинка чорна')  elif a == 'c1' or a == 'c3' or a == 'c5' or a == 'c7':  print('Клітинка чорна')  elif a == 'c2' or a == 'c4' or a == 'c6' or a == 'c8':  print('Клітинка біла')  elif a == 'd1' or a == 'd3' or a == 'd5' or a == 'd7':  print('Клітинка біла')  elif a == 'd2' or a == 'd4' or a == 'd6' or a == 'd8':  print('Клітинка чорна')  elif a == 'e1' or a == 'e3' or a == 'e5' or a == 'e7':  print('Клітинка чорна')  elif a == 'e2' or a == 'e4' or a == 'e6' or a == 'e8':  print('Клітинка біла')  elif a == 'f1' or a == 'f3' or a == 'f5' or a == 'f7':  print('Клітинка біла')  elif a == 'f2' or a == 'f4' or a == 'f6' or a == 'f8':  print('Клітинка чорна')  elif a == 'g1' or a == 'g3' or a == 'g5' or a == 'f7':  print('Клітинка чорна')  elif a == 'g2' or a == 'g4' or a == 'g6' or a == 'f8':  print('Клітинка біла')  elif a == 'h1' or a == 'h3' or a == 'h5' or a == 'h7':  print(біла)  elif a == 'h2' or a == 'h4' or a == 'h6' or a == 'h8':  print('Кл чорна')  else:  print('Не правельний ввод')  printTimeStamp('\nМоскаленко, Яковенко \n')  input('\n') | |  |
| 19. | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  mark = input('Введіть буквенну оцінку(від A до F): ')  if mark == 'A+' :  print('Оцінка більше 4 балів')  elif mark == 'A' :  print('Ваша оцінка становить: 4.0 бала')  elif mark == 'A-' :  print('Ваша оцінка становить: 3.7 бала')  elif mark == 'B+' :  print('Ваша оцінка становить: 3.3 бала')  elif mark == 'B' :  print('Ваша оцінка становить: 3.0 бала')  elif mark == 'B-' :  print('Ваша оцінка становить: 2.7 бала')  elif mark == 'C+' :  print('Ваша оцінка становить: 2.3 бала')  elif mark == 'C' :  print('Ваша оцінка становить: 2.0 бала')  elif mark == 'C-' :  print('Ваша оцінка становить: 1.7 бала')  elif mark == 'D+' :  print('Ваша оцінка становить: 1.3 бала')  elif markh == 'D' :  print('Ваша оцінка становить: 1.0 бал')  elif mark == 'F' :  print('Ваша оцінка становить: 0 балов')  else:  print('Помилка, введіть буквенну оцінку від A до F') | | 2 бали |
| 20. | 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.  import datetime  def printTimeStamp(name):  print('Автори програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко")  a = 0  b = 0  while True:  mark = float(input("Введіть оцінку: "))  a += 1  b += mark  if mark > 4:  print("Оцінка: А+")  elif mark == 4:  print("Оцінка : А")  elif mark == 3.7:  print("Оцінка : A-")  elif mark == 3.3:  print("Оцінка : B+")  elif mark == 3:  print("Оцінка : B")  elif mark == 2.7:  print("Оцінка : B-")  elif mark == 2.3:  print("Оцінка : C+")  elif mark == 2:  print("Оцінка : C")  elif mark == 1.7:  print("Оцінка : C-")  elif mark == 1.3:  print("Оцінка : D+")  elif mark == 1:  print("Оцінка : D")  elif mark == 0:  print("Оцінка : D-")  elif mark == -1:  a -= 1  b -= mark  break  else:  a -= 1  b -= mark  print("Такої оцінки немає")  c =(b/a)\*10  if c in range(0,9):  print("Буквений варіант: F")  elif c in range(10, 12):  print("Буквений варіант: D")  elif c in range(13, 16):  print("Буквений варіант: D+")  elif c in range(17, 19):  print("Буквений варіант: C-")  elif c in range(20, 22):  print("Буквений варіант: C")  elif c in range(23, 26):  print("Буквений варіант: C+")  elif c in range(27, 29):  print("Буквений варіант: B-")  elif c in range(30, 32):  print("Буквений варіант: B")  elif c in range(33, 36):  print("Буквений варіант: B+")  elif c in range(37, 39):  print("Буквений варіант: A-")  elif c == 40:  print("Буквений варіант: A")  elif c > 40:  print("Буквений варіант: A+")  print("Середня оцінка: ")  d = c/10  print(d) | | 2 бали  В19 |
| 22. | Електромагнітне випромінювання можна розділити на 7 категорій за частотою:    Напишіть програму, яка зчитує частоту радіації та виводить відповідну назву хвиль.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  A = input('Введть частоту радiаiї: ')  a = int(A)  if a >= 0 and a < 3 \* 10\*\*9:  print('\nIt\'s radio waves \n')  elif a > 3 \* 10\*\*9 and a < 3 \* 10\*\*12:  print('\nMicrowaves\n')  elif a > 3 \* 10\*\*12 and a < 4.3 \* 10\*\*4:  print('\nInfrared light\n')  elif a > 4.3 \* 10\*\*14 and a < 7.5 \* 10\*\*14:  print('\nVisible light\n')  elif a > 7.5 \* 10\*\*14 and a < 3 \* 10\*\*17:  print('\nUltraviolet light\n')  elif a > 3 \* 10\*\*17 and a < 3 \* 10\*\*19:  print('\nX-rays\n')  elif a > 3 \* 10\*\*19:  print('\nGamma rays\n')  else:  print('Не вiрний ввод')  printTimeStamp('\nМоскаленко, Яковенко\n') | | 2 бали |
| 24. | 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.    import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  A = input('Ведіть рівень шуму в децебелах: ')  a = int(A)  if a < 40:  print('\nNoise level less then quiet room\n')  elif a == 40:  print('\nNoise level is Quiet level\n')  elif a > 40 and a < 70:  print('\nNoise level less then Alarm clock and more then Quiet room\n')  elif a == 70:  print('\nNoise level is Alarm clock\n')  elif a > 70 and a < 106:  print('\nNoise level less then Gas lawnmower and more then Alarm clock\n')  elif a == 106:  print('\nNoise level is Gas lawnmower\n')  elif a > 106 and a < 130:  print('\nNoise level less then Jackhammer and more then Gas lawnmower\n')  elif a == 130:  print('\nNoize level is Jackhammer\n')  else:  print('\nNoise level more than any\n')  printTimeStamp('\nМоскаленко, Яковенко\n')  input('\n') | | 2 бали |
| 25. | Напишіть програму, яка відобразить табличку множення 10х10, як показано на рисунку: | | 2 бали |
| 26. | 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:    import datetime  def factorise(n):  factors = []  factor = 2  if n < factor:  print("Помилка")  return 0  else:  while( factor \* factor <= n) and (n >= 2):  if n % factor == 0:  factors.append(factor)  n = n // factor  else:  factor = factor + 1  factors.append(n)  print("Фактори числа", n, ":", factors)  n = int(input("Введiть число(бiльше 2) : "))  factorise(n)  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp("Москаленко, Яковенко") | | 2 бали |
| 27. | Виведіть на екран символи таблиці ASCII від 0 до 127 разом з їх номером у десятковій, двійковій, вісімковій та шістнадцятковій системах числення.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  for i in range(0, 128):  print(chr(i), end=' ')  printTimeStamp('\nМоскаленко, Яковенко\n')  input('\n') | | 2 бали |

**Рівень В**

**Таблиця дня 1 частини В**

|  |  |
| --- | --- |
| **Рівень В** | **Бали** |
| **1В1** | **2** |
| **1В2** | **---** |
| **1В3** | **2** |
| **1В4** | **2,5** |
| **1В5** | **2,5** |
| **1В6** | **2,5** |
| **1В7** | **2,5** |
| **1В8** | **2,5** |
| **1В9** | **2,5** |
| **1В10** | **3** |
| **1В11** | **2** |
| **1В12** | **2,5** |
| **1В13** | **2** |
| **1В14** | **---** |
| **1В15** | **--** |
| **1В16** | **2,5** |
| **1В17** | **2,5** |
| **1В18** | **2,5** |
| **1В19** | **2** |
| **1В20** | **2** |
| **1В21** | **2** |
| **1В22** | **2** |
| **1В23** | **2,5** |
| **1В24** | **---** |
| **1В25** | **2** |
| **1В26** | **2** |
| **1В27** | **2** |
| **1В28** | **2** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Сума В** | **57** |

**Рівень С**

|  |  |  |
| --- | --- | --- |
| **№** | **Умова задачі** | **Бали /**  **Зв’язки** |
| 1. | 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.  import datetime, random  printTimeStamp = lambda x: print('Автор програми: {0}\nЧас компіляції: {1}'.format(x, str(datetime.datetime.now())))  allCell = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, "0", "00"]  reds = [1, 3, 5, 7, 9, 12, 14, 16, 18, 19, 21, 23, 25, 27, 30, 32, 34, 36]  blacks = [2, 4, 6, 8, 10, 11, 13, 15, 17, 20, 22, 24, 26, 28, 29, 31, 33, 35]  def oneNumber(num):  win = random.choice(allCell)  if (win == "0" or win == "00") and num == win:  return win  else:  print("На рулетці випало", win)  if win == int(num):  print("Виплатити", win)  else:  print("Ви програли")  def redVSblack(color):  win = random.choice(allCell)  print("На рулетці випало", "Red" if win in reds else "Black" if win in blacks else win)  print("Виплатити Red") if ((win in reds) and color == "Red") else print("Виплатити Black") if (win in blacks) and color == "Black" else print("Ви програли")  def oddEven(choice):  win = random.randint(1, 36)  print("На рулетці випало", win)  if ((win % 2 == 0) and choice == "Even"):  print("Виплатити Even")  elif (win % 2 != 0) and choice == "Odd":  print("Виплатити Odd")  else:  print("Ви програли")  def inHalf(choice):  win = random.choice(allCell)  print("На рулетці випало", win)  if (win <= 18 and choice == "First"):  print("Виплатити 1 to 18")  elif win >= 19 and choice == "Second":  print("Виплатити 19 to 36")  else:  print("Ви програли")  while True:  betOneNumber = oneNumber(input("\n{}\nВведіть число для ставки 'Одне число' (від 1 до 36, 0 або 00): ".format("="\*15)))  if betOneNumber != None:  print("Виплатити", betOneNumber)  continue  betRedVsBlack = redVSblack(input("Введіть колір для ставки 'Червоне vs Чорне' (Red або Black): "))  betOddEven = oddEven(input("Введіть вибір для ставки 'Парне vs Непарне' (Odd або Even): "))  betInHalf = inHalf(input("Введіть вибір для ставки 'Від 1 до 18 vs від 19 до 36' (First або Second): "))  printTimeStamp("Москаленко, Яковенко") | 4 бали |
|  |  |  |
| 2. | 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!  year = int(input("Введіть рік: "))  if (year % 400 == 0):  leap\_year = True  elif (year % 100 == 0):  leap\_year = False  elif (year % 4 == 0):  leap\_year = True  else:  leap\_year = False  month = int(input("Введіть місяць [1-12]: "))  if month in (1, 3, 5, 7, 8, 10, 12):  month\_length = 31  elif month == 2:  if leap\_year:  month\_length = 29  else:  month\_length = 28  else:  month\_length = 30  day = int(input("Введіть день [1-31]: "))  if day < month\_length:  day += 1  else:  day = 1  if month == 12:  month = 1  year += 1  else:  month += 1  print("Наступна дата [yyyy-mm-dd] %d-%d-%d." % (year, month, day))  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name="Москаленко, Яковенко"  printTimeStamp(name) | 5 балів |
| 3. | 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.  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\*(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 \* (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 \* (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 \* (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  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp('Москаленко, Яковенко') | 5 балів |
| 4. | 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.  tar = input("Введіть назву тарифу( 1000 , 2000 , 5000 ) ")  #1000mb  if tar == "1000":  mb\_status = float(input("Введіть кількість викорастаних МБ: "))  if mb\_status < 0:  print("Input Error")  elif mb\_status <= 1000:  print("Ваш рахунок: 20 ГРН ")  elif mb\_status > 1000:  out\_tar = (mb\_status - 1000) \* 0.05  all\_cost = 20 + out\_tar  print("Ваш рахунок: " + str(all\_cost) + " ГРН" )  print("Тариф 2000: 2000МБ + 0.4ГРН позатарфино за 1МБ \nТариф 5000: 5000МБ + 0.2ГРН позатарфино за 1МБ")  #2000mb  if tar == "2000":  mb\_status = float(input("Введіть кількість викорастаних МБ: "))  if mb\_status < 0:  print("Input Error")  elif mb\_status <= 2000:  print("Ваш рахунок: 35 ГРН ")  elif mb\_status > 2000:  out\_tar = (mb\_status - 2000) \* 0.04  all\_cost = 35 + out\_tar  print("Ваш рахунок: " + str(all\_cost) + " ГРН" )  print("Тариф 2000: 2000МБ + 0.4ГРН позатарфино за 1МБ \nТариф 5000: 5000МБ + 0.2ГРН позатарфино за 1МБ")  #5000mb  if tar == "5000":  mb\_status = float(input("Введіть кількість викорастаних МБ: "))  if mb\_status < 0:  print("Input Error")  elif mb\_status <= 5000:  print("Ваш рахунок: 85 ГРН ")  elif mb\_status > 5000:  out\_tar = (mb\_status - 5000) \* 0.02  all\_cost = 85 + out\_tar  print("Ваш рахунок: " + str(all\_cost) + " ГРН" )  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp('Москаленко, Яковенко') | 4 бали |
| 5. | 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.  import random, datetime  point = 0  trys = 1  while trys == 1:  trys -= 1  kubiki= [random.randint(1,6) for i in range(6)]  print(kubiki)  point += kubiki.count(1) \* 100  point += kubiki.count(5) \* 50  break\_group = 0  for i in range (1,7):  if kubiki.count(i) >=3:  print("{} однакові кості '{}' +{}".format(kubiki.count(i), i, (kubiki.count(i)- 2) \* 100 \*(10 if i == 1 else i)))  point += (kubiki.count(i)-2) \* 100 \*(10 if i == 1 else 1)  break\_group += 1  if len(set(kubiki)) == 3 and break\_group == 0:  print("Три пари +750 та ще одна спроба")  point += 750  trys += 1  if len(set(kubiki)) == 6:  print("Три пари +1500 та ще одна спроба")  point += 1500  trys +=1  print("Ваші очки", point)  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp('Москаленко, Яковенко') | 5 балів |

**Таблиця дня 1 частини C**

|  |  |
| --- | --- |
| **Рівень С** | **Бали** |
| **1С1** | **4** |
| **1С2** | **5** |
| **1С3** | **5** |
| **1С4** | **4** |
| **1С5** | **5** |
| **1С6** | **--** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **Сума С** | **23** |

**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** |  |  |
| **1А21** | **1,5** | **1В21** | **2** |  |  |
|  |  | **1В22** | **2** |  |  |  |
|  |  | **1В23** | **2,5** |  |  |  |
|  |  | **1В24** | **2** |  |  | **Сума** |
|  |  | **1В25** | **2** |  |  | **98.5** |
|  |  | **1В26** | **2** |  |  |  |
|  |  | **1В27** | **2** |  |  |  |
|  |  | **1В28** | **2** |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Сума А** | **18.5** | **Сума В** | **57** | **Сума С** | **23** |  |

День другий

Рівень А

|  |  |  |
| --- | --- | --- |
| 1. | 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.  b = [] print("Щоб видалити 3 найменших і найбільших числа введіть '+'\nВведіть число щоб додати його в список :") while True:  a = input()  if a == '+' and len(b)>=6:  b.sort()  for x in range(3):  b.pop()  b.reverse()  b.pop()  print(b)  elif a.isdigit():  b.append(int(a))  else:  print("Ви ввели щось не те!") import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  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.  k = [] o = [] m = [] while True:  a = input('Введіть число: ')  if a != '':  a = float(a)  k.append(a)  else:  break c = sum(k) / len(k) for i in range(len(k)):  if k[1] < c:  o.append(k[i])  elif k[i] >= c:  m.append(k[i])  else:  print('') print('Список чисел, які менше за середнє значення: {}'.format(o)) print('Список чисел, які більше або рівні середньому значенню: {}'.format(m)) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко " printTimeStamp(name)  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  list1 = [] while True:  a = input('Введіть слово: ')  if a == '':  break  else:  list1.append(a) list2 = [] for i in list1:  if i not in list2:  list2.append(i) print(list2) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко " printTimeStamp(name)  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".  import random article = ['the', 'a', 'one', 'some', 'any'] noun = ['boy', 'girl', 'dog', 'town', 'car'] verb = ['drove', 'jumped', 'ran', 'walked', 'skipped'] preposition = ['to', 'from', 'over', 'under', 'on'] for i in range(10):  print('{} {} {} {} {} {}.'.format(  article[random.randint(0,4)].capitalize(),  noun[random.randint(0,4)],  verb[random.randint(0,4)],  preposition[random.randint(0,4)],  article[random.randint(0,4)],  noun[random.randint(0,4)])) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  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)  import winsound m = [659, 659, 659, 523, 659, 784,  392, 523,392, 330, 440, 494,  466, 440, 392, 659, 784, 880,  698, 784, 659, 523, 587, 494] n = [250, 250, 300, 250, 250, 300,  300, 275, 275,275, 250, 250,  275, 275, 275, 250, 250, 275,  275, 225, 250, 250, 225, 225] for x in range(len(m)):  winsound.Beep(m[x],n[x]) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  Вивід: | 1 бал |
| 6. | 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)  import winsound m = [[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 x in m:  print(x)  winsound.Beep(x[0],x[1]) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко " printTimeStamp(name)  Вивід: | 1 бал |
| 7. | 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.  k = [] l = ['s', 'h', 'd', 'c'] s = ['J', 'Q', 'K', 'A'] for i in range(len(l)):  for a in range(2, 11):  b = str(a) + str(l[i])  k.append(b)  for c in range(len(s)):  b = str(s[c]) + str(l[i])  k.append(b) print(k) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  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.  from random import randrange def sattoloCycle(items):  i = len(items)  while i > 1:  i = i-1  j = randrange(i)  aa = k.pop(i)  bb = k.pop(j)  k.insert(j, aa)  k.insert(i, bb)  return k = [] l = ["s","h","d","c"] s = ["J","Q","K","A"] for i in range(len(l)):  for a in range(2,11):  b = str(a) + str(l[i])  k.append(b)  for c in range(len(s)):  b = str(s[c]) + str (l[i])  k.append(b) print(k) print("\n") sattoloCycle(list(k)) print(k) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  Вивід: | 1 бал |
| 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.  from random import randrange def sattoCycle(items):  i = len(items)  while i>1:  i=i-1  j= randrange(i)  aa =k.pop(i)  bb =k.pop(j)  k.insert(j,aa)  k.insert(i,bb)  return k=[] l=["s","h","d","с"] s=["J","Q","K","A"] for i in range(len(l)):  for a in range(2,11):  b=str(a)+str(l[i])  k.append(b)  for c in range(len(s)):  b=str(s[c])+str(l[i])  k.append(b) sattoCycle(list(k)) pq = int(input("Кількість гравців:")) cq = int(input("Кількість карт на гравця:")) playerlist =[] cardlist=[] cardlistt=[] for i in range(pq):  playerlist.append("Гравець{}".format(i)) if len(k) <(cq\*pq):  print("Error") else:  for kkk in range(len(playerlist)):  for kkkk in range(cq):  k.pop(kkkk)  cardlistt.append(k[kkkk])  cardlist.append(list(cardlistt))  cardlistt.clear()  for i in range(len(playerlist)):  print("{0}:{1}".format(playerlist[i],cardlist[i])) import datetime def printTimeStamp(name) :   print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  Вивід:    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.  import datetime, random  n = 1000 def first\_kind\_generator(n):  for i in range(1, n + 1):  p = random.randint(1, 6) + random.randint(1, 6)  i += 1  yield p l = [] for i in first\_kind\_generator(n):  l.append(i) print("{:8}{:8}".format('Total','Simulated %')) for i in range(2,13):  print("{:2}{:12}".format(i, l.count(i)/10))  i += 1    printTimeStamp = lambda x: print('Автор програми: {0}\nЧас компіляції: {1}'.format(x, str(datetime.datetime.now()))) printTimeStamp("Яковенко, Москаленко")  Вивід: | 1 бал |

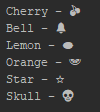
**Рівень В**

|  |  |  |  |
| --- | --- | --- | --- |
| 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.  holy = {'A':'Newfoundland',  'B':'Nova Scotia',  'C':'Prince Edward Island',  'E':'New Brunswick',  'G':'Quebec',  'H':'Quebec',  'J':'Quebec',  'K':'Ontario',  'L':'Ontario',  'M':'Ontario',  'N':'Ontario',  'P':'Ontario',  'R':'Manitoba',  'S':'Saskatchewan',  'T':'Alberta',  'V':'British Columbia',  'W':'Nunavut або Nortwest Territories',  'Y':'Yukon'} while True:  try:  code = list(input('Введіть код: '))  code[1] = int(code[1])  code[3] = int(code[3])  code[5] = int(code[5])  except ValueError:  print('Некоректний код')  continue  if len(code) == 6:  break  else:  print('Некоректний код')  continue for i in holy:  if i == code[0]:  print(holy.get(i))  else:  continue if code[1] == 0:  print('Село') else:  print('Місто') import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  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.  l1 = [".","A","a","D","d","G","g","J","j","M","m","P","p","T","t","W","w"] l2 = [",","B","b","E","e","H","h","K","k","N","n","Q","q","U","u","X","x"] l3 = ["?","C","c","F","f","I","i","L","l","O","o","R","r","V","v","Y","y"] l4 = ["!",None,None,None,None,None,None,None,None,"S","s",None,None,"Z","z"] l5 = [":"] LL = [l1,l2,l3,l4,l5] while True:  a = list(input("Введіть щось :"))  s = ""  for x in a:  if x == " ":  s += "0"  for xx in LL:  try:  s+=(str(((xx.index(x)+1)//2)+1)\*(LL.index(xx)+1))  except:  pass  print(s) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко, Москаленко" printTimeStamp(name)  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 .  a = int(input('Введіть число:')) b = 0 lis = [] for i in range(2,a+1):  lis.append(i) for z in lis:  p = lis[b]  b += 1  if z\*\*2 >= a:  break  for i in lis:  if i % p == 0 and i != p:  lis.remove(i) print(lis) import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко Москаленко" printTimeStamp(name)  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.  import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) score = {"a": 1 , "b": 3 , "c": 3 , "d": 2 ,  "e": 1 , "f": 4 , "g": 2 , "h": 4 ,  "i": 1 , "j": 8 , "k": 5 , "l": 1 ,  "m": 3 , "n": 1 , "o": 1 , "p": 3 ,  "q": 10, "r": 1 , "s": 1 , "t": 1 ,  "u": 1 , "v": 4 , "w": 4 , "x": 8 ,  "y": 4 , "z": 10} def Scrable(a):  start = 0  for i in a:  i = i.lower()  start = start + score[i]  return start a = input("Введіть своє слово: ") print(Scrable(a)) printTimeStamp('\Яковенко Москаленко\n') input('\n')  Вивід:    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.  import sys num = int(input("Enter number(0-10000):")) text = [] if num <= 0 or num >= 10000:  print("error")  sys.exit(0) t = {'0':"",'1':"одна тисяча",'2':"дві тисячі",'3':"три тисячі",'4':"чотири тисячі",'5':"п'ять тисяч",'6':"шість тисяч",'7':"сім тисяч",'8':"вісім тисяч",'9':"дев'ять тисяч"} s = {'0':"",'1':"сто",'2':"двісті",'3':"триста",'4':"чотириста",'5':"п'ятсот",'6':"шістсот",'7':"сімсот",'8':"вісімсот",'9':"дев'ятсот"} d = {'0':"",'1':"десять",'2':"двадцять",'3':"тридцять",'4':"сорок",'5':"п'ятьдесят",'6':"шістдесят",'7':"сімдесят",'8':"вісімдесят",'9':"дев'яносто"} o = {'0':"",'1':"одна",'2':"дві",'3':"три",'4':"чотири",'5':"п'ять",'6':"шість",'7':"сім",'8':"вісім",'9':"дев'ять"} num1 = list(str(num)) if len(num1) == 4:  if num1[0] in t.keys():  text.append(t[num1[0]])  if num1[1] in s.keys():  text.append(s[num1[1]])  if num1[2] in d.keys():  text.append(d[num1[2]])  if num1[3] in o.keys():  text.append(o[num1[3]]) elif len(num1) == 3:  if num1[0] in s.keys():  text.append(s[num1[0]])  if num1[1] in d.keys():  text.append(d[num1[1]])  if num1[2] in o.keys():  text.append(o[num1[2]]) elif len(num1) == 2:  if num1[0] in d.keys():  text.append(d[num1[0]])  if num1[1] in o.keys():  text.append(o[num1[1]]) elif len(num1) == 1:  if num1[0] in o.keys():  text.append(o[num1[0]]) print(" ".join(text)+" грн")  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко Москаленко" printTimeStamp(name)  Вивід: | 3 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  import datetime name = "Яковенко Москаленко" def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) printTimeStamp(name) a = int(input("Введіть клькість пасажирів: ")) print("Введіть прізвище, дату здачі та дату видачі багажу") if a<=12:  c = []  for i in range(1,a+1):  b = input()  b = b.split(" ")  c.append(b[0]+" "+str(i)+" \n")  print("".join(c)) else:  print("Вас забагато!")  Вивід: | 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.  def precedence(a):  if a=="+"or a == "-":  return 1  elif a=="/"or a == "\*":  return 2  elif a=="^":  return 3  else:  return -1,"Не введено оператора" while True:  if \_\_name\_\_ == "\_\_main\_\_":  print(precedence(input("Введіть знак: "))) import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко Москаленко"  Вивід: | 2 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].  i = 0 a = input("Введіть IPv4: ") if a.count(".")!=4:  i+=5 b = a.split(".") del(b[-1]) for x in b:  try:  if 0<=int(x)<=225:  i+=1  except:  i+=5 if i ==4:  print("Bірний ІР") else:  print("Невірний ІР") import datetime def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now())) name="Яковенко Москаленко" printTimeStamp(name)  Вивід: | |  |

**Рівень С**

|  |  |
| --- | --- |
| 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.  import random, sys, datetime printTimeStamp = lambda x: print('Автор програми: {0}\nЧас компіляції: {1}'.format(x, str(datetime.datetime.now()))) p1 = input("Enter first player name: ") p2 = input("Enter second player name: ") class Kard:  def f(self):  lst = [x for x in range(1, self.amount + 1)]  random.shuffle(lst)  for i, y in enumerate(lst):  print('{:\*^25}'.format('\*'))  yield y   def \_\_init\_\_(self, amount):  self.amount = amount  self.gen = self.f()  class Loto:  def set\_card(self):  num = set()  while len(num) < self.all\_row \* 5:  num.add(random.randint(1, 91))  cards = list(num)  random.shuffle(cards)   while len(cards) % self.all\_row != 0:  cards.append('None')  self.all\_row = int(len(cards) / self.all\_row)  cards = [cards[i: i + self.all\_row] for i in range(0, len(cards), self.all\_row)]   for i in range(len(cards)):  cards[i].sort()  self.card\_user = cards[:3]  self.card\_comp = cards[3:]   def \_\_init\_\_(self, amount\_card):  row = 3  self.all\_row = row \* amount\_card  self.set\_card()  class Player(Loto):  def \_\_init\_\_(self, name):  self.name = name  self.score = 0  def get\_card(self, card\_player):  print('{:-^25}'.format(self.name))  print('{0[0]:>2} {0[1]:<10} {0[2]:<5} {0[3]} {0[4]} '.format(card\_player[0]))  print('{0[0]:>4} {0[1]:<6} {0[2]:<4} {0[3]:<4} {0[4]} '.format(card\_player[1]))  print('{0[0]} {0[1]:<5} {0[2]:<5} {0[3]:<5} {0[4]} '.format(card\_player[2]))  print('{:-^25}'.format('-'))   def search(self, card\_player, num\_cask):  for i, n in enumerate(card\_player):  if num\_cask in n:  card\_player[i][n.index(num\_cask)] = '0'  self.score += 1  if self.score == 15:  print('{} win!'.format(self.name))  sys.exit(1)  return True def main():  game = Loto(2)  kard = Kard(90)  player1 = Player(p1)  player2 = Player(p2)   while True:  num\_cask = next(kard.gen)  player1.get\_card(game.card\_user)  player2.get\_card(game.card\_comp)  if player1.search(game.card\_user, num\_cask):  continue  elif player2.search(game.card\_comp, num\_cask):  continue  if \_\_name\_\_ == '\_\_main\_\_':  main() printTimeStamp("Яковенко Москаленко")  Вивід: |

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",

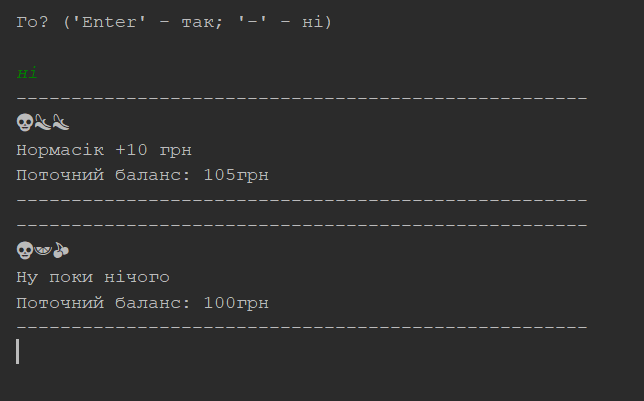
Star - u"\u2606",

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.

import random, time  
money = 100  
lich = 0  
combin = [u"\U0001F352", u"\U0001F514", u"\U0001F34C", u"\U0001F34A", u"\u2606", u"\U0001F480"]  
print("\nГо? ('Enter' - так; '-' - нi)")  
while True:  
 if money < 5:  
 print("На вашому рахунку недостатньо коштiв\nНе повезло, не фартануло!")  
 break  
 g = input()  
 if g == "-":  
 break  
 money -= 5  
 r1 = random.choice(combin)  
 r2 = random.choice(combin)  
 r3 = random.choice(combin)  
 print("-" \* 52)  
 print(r1, end="")  
 time.sleep(0.9)  
 print(r2, end="")  
 time.sleep(1.2)  
 print(r3)  
 time.sleep(0.6)  
  
 if r1 == r2 == r3 == u"\U0001F514":  
 money += 100  
 print("Ну ти мочиш, дядя +100 грн")  
 elif r1 == r2 == r3 != u"\U0001F480":  
 money += 25  
  
  
 print("Опачки ізі +25 грн")  
 elif r1 == r2 != u"\U0001F480" or r1 == r3 != u"\U0001F480" or r2 == r3 != u"\U0001F480":  
 money += 10  
 print("Нормасік +10 грн")  
 elif r1 == r2 == r3 == u"\U0001F480":  
 money -= money  
 print("Ну ти ...... усе провтичив")  
 elif r1 == r2 == u"\U0001F480" or r1 == r3 == u"\U0001F480" or r2 == r3 == u"\U0001F480":  
 money -= 5  
 print("Мiнус 5 грн")  
 else:  
 print("Ну поки нiчого")  
 print('Потoчний баланс: {}грн'.format(money))  
 print("-"\*52)  
 time.sleep(0.5)

Вивід:



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

Студентки групи 1П-17 Москаленко В.І.

**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  def gcd(x,y):  if (x == y):  return y  if (x < y):  return gcd(x, y-x)  if (x > y):  return gcd(x-y, y)  x = 98  y = 56  if(gcd(x, y)):  print('НСД', x, 'і', y, 'є', gcd(x, y))  else:  print('Не знайдено')  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name="Москаленко Яковенко"  printTimeStamp(name) | 1 бал |
| 2. | Вивести n-те число Каталана за допомогою рекурсивного алгоритму  def catalan(n):  if n >= 2:  c = ((2 \* ((2\*n) - 1)) / (n+1)) \* catalan(n-1)  return int(c)  return 1  print(catalan(0))  print(catalan(1))  print(catalan(2))  print(catalan(3))  print(catalan(4))  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name=" Москаленко Яковенко "  printTimeStamp(name) | 1 бал |
| 3. | Опишіть рекурсивну функцію для піднесення числа до цілої додатної степені  def expt(b, n):  if n==0:  return 1  return b\*expt(b, n-1)  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name=" Москаленко Яковенко "  printTimeStamp(name) | 1 бал |
| 4. | Більш ефективне рекурсивне визначення , що охоплює випадок  import datetime  name = " Москаленко Яковенко "  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp(name)  def a(x, n):  if n==0:  return 1  elif n > 0 and n%2==0:  return (x\*\*(n/2))\*\*2  elif n>0 and n%2!=0:  return(x\*x\*\*(n-1))  elif n<0:  return 1/(x\*\*n)  print(a((int(input("Введіть число: "))),(int(input("Введіть степень: ")))) | 1 бал |
| 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  import datetime  name = " Москаленко Яковенко "  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  printTimeStamp(name)  n = int(input("Введвть число :"))  def g(n):  if n ==1:  print(n)  elif n % 2 == 0:  print(n)  return(g(n/2))  elif n % 2 != 0:  print(n)  return(g((3\*n)+1))  g(n) | 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.  prompt = "Введіть число: "  sum = 0  count = 1  while True:  s = input(prompt)  prompt = "Введіть інше число : "  if s.lower() == '':  break  try:  sum += float(s)  print ("Ви ввели %s чисел сума яких %s." % (count, sum))  count += 1  except ValueError:  print( "Неправильний ввід")  print ("Ви ввели %s чисел сума яких %s." % (count, sum))  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name=" Москаленко Яковенко "  printTimeStamp(name) | 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  import math  coef = 1  rows = int(input('Введiть цифру, вона вказуватиме програмi розмiр пiрамiди :'))  stringVal = ""  for i in range(rows):  for space in range(1,rows - i + 1):  stringVal = stringVal + " "  for j in range(0,i + 1):  if(i == 0 or j == 0):  coef = 1  else:  coef = coef \* (i - j + 1) / j  temp = coef  TotalSpace=0  while(temp != 0):  TotalSpace = TotalSpace + 1  temp = int(math.floor(temp / 10))  p=0  while((p+TotalSpace)!=4):  stringVal = stringVal + " "  p=p+1  stringVal = stringVal + str(int(math.floor(coef)))  stringVal = stringVal + "\n"  print(stringVal)  printTimeStamp('\n Москаленко Яковенко \n')  input('\n') | 2,5 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.  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  ALPHA = "".join(map(chr, range(ord(" "), ord("я") + 1)))  def encode(text, step):  return text.translate(  str.maketrans(ALPHA, ALPHA[step:] + ALPHA[:step]))  b = input('Введіть значення ссуву: ')  a = input('Ввод: ')  print(encode(a, int(b)))  printTimeStamp('\n Москаленко Яковенко \n')  input('\n') | 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  **.... . .-.. .-.. --- .-- --- .-. .-.. -..**  CODE = {'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': '----.'  }  def main():  msg = str(input('MESSAGE: '))  for char in msg:  print (CODE[char.upper()],)  if \_\_name\_\_ == "\_\_main\_\_":  main()  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name=" Москаленко Яковенко "  printTimeStamp(name) | 3 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.  import random  def drawBoard(board):  print(' | |')  print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])  print(' | |')  print('-----------')  print(' | |')  print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])  print(' | |')  print('-----------')  print(' | |')  print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])  print(' | |')  def inputPlayerLetter():  letter = ''  while not (letter == 'X' or letter == 'O'):  print('Do you want to be X or O?')  letter = input().upper()  if letter == 'X':  return ['X', 'O']  else:  return ['O', 'X']  def whoGoesFirst():  if random.randint(0, 1) == 0:  return 'computer'  else:  return 'player'  def playAgain():  print('Do you want to play again? (yes or no)')  return input().lower().startswith('y')  def makeMove(board, letter, move):  board[move] = letter  def isWinner(bo, le):  return ((bo[7] == le and bo[8] == le and bo[9] == le) or  (bo[4] == le and bo[5] == le and bo[6] == le) or  (bo[1] == le and bo[2] == le and bo[3] == le) or  (bo[7] == le and bo[4] == le and bo[1] == le) or  (bo[8] == le and bo[5] == le and bo[2] == le) or  (bo[9] == le and bo[6] == le and bo[3] == le) or  (bo[7] == le and bo[5] == le and bo[3] == le) or  (bo[9] == le and bo[5] == le and bo[1] == le))  def getBoardCopy(board):  dupeBoard = []  for i in board:  dupeBoard.append(i)  return dupeBoard  def isSpaceFree(board, move):  return board[move] == ' '  def getPlayerMove(board):  move = ' '  while move not in '1 2 3 4 5 6 7 8 9'.split() or not isSpaceFree(board, int(move)):  print('What is your next move? (1-9)')  move = input()  return int(move)  def chooseRandomMoveFromList(board, movesList):  possibleMoves = []  for i in movesList:  if isSpaceFree(board, i):  possibleMoves.append(i)  if len(possibleMoves) != 0:  return random.choice(possibleMoves)  else:  return None  def getComputerMove(board, computerLetter):  if computerLetter == 'X':  playerLetter = 'O'  else:  playerLetter = 'X'  for i in range(1, 10):  copy = getBoardCopy(board)  if isSpaceFree(copy, i):  makeMove(copy, computerLetter, i)  if isWinner(copy, computerLetter):  return i  for i in range(1, 10):  copy = getBoardCopy(board)  if isSpaceFree(copy, i):  makeMove(copy, playerLetter, i)  if isWinner(copy, playerLetter):  return i  move = chooseRandomMoveFromList(board, [1, 3, 7, 9])  if move != None:  return move  if isSpaceFree(board, 5):  return 5  return chooseRandomMoveFromList(board, [2, 4, 6, 8])  def isBoardFull(board):  for i in range(1, 10):  if isSpaceFree(board, i):  return False  return True  print('Welcome to Tic Tac Toe!')  while True:  theBoard = [' '] \* 10  playerLetter, computerLetter = inputPlayerLetter()  turn = whoGoesFirst()  print('The ' + turn + ' will go first.')  gameIsPlaying = True  while gameIsPlaying:  if turn == 'player':  drawBoard(theBoard)  move = getPlayerMove(theBoard)  makeMove(theBoard, playerLetter, move)  if isWinner(theBoard, playerLetter):  drawBoard(theBoard)  print('Hooray! You have won the game!')  gameIsPlaying = False  else:  if isBoardFull(theBoard):  drawBoard(theBoard)  print('The game is a tie!')  break  else:  turn = 'computer'  else:  move = getComputerMove(theBoard, computerLetter)  makeMove(theBoard, computerLetter, move)  if isWinner(theBoard, computerLetter):  drawBoard(theBoard)  print('The computer has beaten you! You lose.')  gameIsPlaying = False  else:  if isBoardFull(theBoard):  drawBoard(theBoard)  print('The game is a tie!')  break  else:  turn = 'player'  if not playAgain():  break  import datetime  def printTimeStamp(name):  print('Автор програми: ' + name)  print('Час компіляції: ' + str(datetime.datetime.now()))  name=" Москаленко Яковенко "  printTimeStamp(name) | 3 бали |

|  |  |  |
| --- | --- | --- |
| **№** | **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 | 5 балів |

**Рівень С**

import turtle

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 = ['cyan','purple','orange','navy','gold',

'firebrick','lawn green']

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 = [[-200,-100],[0,200],[200,-100]]

sierpinski(myPoints,5,myTurtle)

myWin.exitonclick()

main()

import datetime

def printTimeStamp(name):

print('Автор програми: ' + name)

print('Час компіляції: ' + str(datetime.datetime.now()))

name="Москаленко Яковенко"

printTimeStamp(name)

