Задачи за решаване – Всички задачи да се рашат с функции които връщат резолтат

1. Делене на числа

01\_number\_divide.py

Напишете програма, която приема две числа от потребителя и извършва деление на първото с второто. Обработете възможните грешки, като нулево деление или въвеждане на символи вместо числа. В случай на грешка изведете подходящо съобщение. Подробности: Вашата програма трябва да използва конструкцията **try-except** за обработка на грешките, които могат да възникнат при деленето. Ако второто число е нула или потребителят въведе нечислова стойност, програмата трябва да изведе съобщение за грешка и да продължи с изпълнението си.

**Вход:**

На входа трябва се получават се две цели числа разделени със запетая (**,** ). Опитайте се да разделите първото на второто.

**Изход:**

На изхода трябва да върнете резултата от делението или да прихванете грешката и да върнете някое от следните съобщения

**You can not divide by ZERO**

**You can not divide by symbol**

Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| **6, 2** | **3.0** |
| **6, 0** | **You can not divide by ZERO** |
| **6, a** | **You can not divide by symbol** |

1. Обработка на списък

02\_list\_working.py

Ще получите списък с елементи. Напишете програма, която сумира всички елементи от списъка. Обработете възможните грешки при сумирането и изведете резултата. Вашата програма трябва да използва конструкцията **try-except** за обработка на възможни грешки, като например неправилни типове на елементите в списъка или ако списъка е празен извикайте грешката IndexError и го прихванете. Ако възникне грешка при сумирането, изведете съобщение за грешка.

**Вход:**

На входа се очаква да се получават редица от положителни цели числа.

**Изход:**

На изхода трябва да върнете резултата от сумирането или да прихванете грешката и да върнете някое от следните съобщения

You can not sum int and str!

You can not sum empty list!

Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 4 9 21 81 24 56 | 195 |
| 4 9 21 81 24 56 a | You can not sum int and str! |
|  | You can not sum empty list! |

1. Преобразуване на данни

03\_data\_working.py

Напишете програма, която преобразува въведената от потребителя стойност в цяло число. Обработете възможните грешки при въвеждането на стойността и при опит за преобразуване. Ако преобразуването е успешно, изведете резултата. Вашата програма трябва да използва конструкцията **try-except** за обработка на грешките, които могат да възникнат при въвеждането или преобразуването на стойността. Ако въведената стойност не може да бъде преобразувана в цяло число, изведете съобщение за грешка.

**Вход:**

На входа се очаква да се получи число което да се преобразува в цяло число

**Изход:**

На изхода трябва да върнете резултата от преобразуването или да прихванете грешката и да върнете следното съобщение

You can not make string to integer

Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5.2 | 5 |
| 5 | 5 |
| а | You can not make string to integer |

1. Работа с речник

04\_dictionary\_working.py

Ще получите речник с имена и години на хора. Напишете програма, която приема име и извежда годините на човека. Обработете възможните грешки при достъпването на данните и при въвеждането на неправилно име. Вашият речник може да изглежда като **{'Име1': Години1, 'Име2': Години2, ...}**. Програмата трябва да използва конструкцията **try-except** за обработка на грешки, като например невъведено име или опит за достъпване на несъществуващо име. В случай на грешка изведете съобщение за грешка. В края ако програмата не е влязла в грешка трябва да ипише

That was ages of the persons

**Вход:**

На входа на първи ред ще получите списък от имена и години на човек. Трябва да си добавите в речник.

След това ще получавате имена, които да проверявате на колко години са

**Изход:**

На изхода трябва да върнете резултата годините или да прихванете грешката и да върнете следното съобщение

This person does not exist in this dictionary

Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Стамат 21 Спиридон 11 Кремена 32 Иванка 16 Аспарух 13  Стамат, Спиридон, Кремена | 21, 11, 32  That was ages of the persons |
| Стамат 21 Спиридон 11 Кремена 32 Иванка 16  Аспарух 13  Стамат, Спиридон, Кремена, Кирчо | This person does not exist in this dictionary |

1. Проверка за елемент

05\_search\_element.py

Напишете програма, която проверява дали даден елемент съществува в списък от дадени стойности и го премахвате. Обработете възможните грешки при проверката и изведете подходящ резултаt. Вашият списък може да бъде представен като обикновен списък в Python. Програмата трябва да използва конструкцията **try-except** за обработка на грешки, като например опит за достъпване на несъществуващ елемент. В случай на успех или неуспех изведете подходящо съобщение.

**Вход:**

На входа на първи ред ще получите списък от елементи.

След това ще трябва да получите втори списък, от чиито елементи да проверявате има ли ги в първият списък

**Изход:**

Принтирайте премахнатите елементи или ако прихванете грешка птинтирайте

Element not in the list

Ако не прихванете грешка принтирайте :

Deletion was successful

И във всички случаи принтирайте :

This is my program

Пример

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Стамат 21 Спиридон 11 Кремена 32 Иванка 16 Аспарух 13  Стамат, Спиридон, Кремена | Стамат, Спиридон, Кремена  Deletion was successful  This is my program |
| Стамат 21 Спиридон 11 Кремена 32 Иванка 16  Аспарух 13  Стамат, Спиридон, Кремена, Кирчо | Element not in the list  This is my program |