### **1. Identifying Transaction Type**

**Ramishahope Artificial Intelligence Pvt Ltd**

**36, Old Anandas, SG Arcade, Marudhamalai Main Road, Vadavalli, Coimbatore -641041.**

**+91 6385383227 |** [**www.hopelearning.net**](http://www.hopelearning.net/) **|** [**mdaravind@hopelearning.net**](mailto:mdaravind@hopelearning.net) **| 33AAMCR3722R1ZU**

* Read the input number from user
* If >0, print **(“positive”, “cate: deposit”)**
* eIf >0, print **(“Negative”, “cate: Withdrawal”)**
* Else, print "Zero (No Transaction)”.

### **2. Summing the Digits of a Number**

* Read the input number from user as passcode.
* Sum=0,
* For digit in str(sum): **(here we write string because make the passcode iterate)**
* Sum +=int(digit), print (“sum of digit:”, digit) **(note: sum= sum+ int (digit) )** result will be saved in sum variable in each iterate

### **3. Reversing a Transaction ID**

**Write logic:**

* Trans\_id=int(input("Enter your id")) **from user**
* trans\_str=str(Trans\_id) **convert into string**
* reverse\_id= trans\_str[::-1] **reverse the string**
* Trans\_int=int(reverse\_id) **back to the integer**
* print(Trans\_int) **print**

### **4. Checking if a Number is Prime**

**Write logic: what is- prime the number which can be divisible only 1 and itself**

* **num=int(input(“Enter the number”)**
* If the number < 2, print "Not Prime". **(Note: only even prime number is 2)**
* For I in range(2, (num\*\*0.5)+1):
  + **If num%i==0:, print "Not Prime" and exit.**
* Else:, print "Prime".

**Ramishahope Artificial Intelligence Pvt Ltd**

**36, Old Anandas, SG Arcade, Marudhamalai Main Road, Vadavalli, Coimbatore -641041.**

**+91 6385383227 |** [**www.hopelearning.net**](http://www.hopelearning.net/) **|** [**mdaravind@hopelearning.net**](mailto:mdaravind@hopelearning.net) **| 33AAMCR3722R1ZU**

### **5. Finding the Factorial Using Recursion**

**Write logic:**

* To use recursion we should use function **(def factorial(n):)**
* If n==0 or n==1: return 1 **(because 0! or 1! = 1)**
* Else, return n\*factorial(number - 1). **(n!=n\*n-1!)**
* Print the result.

### **6. Checking if a Number is an Armstrong Number**

**Write logic:**

**Note: Armstrong number is the sum of base number = power of all individual number**

**Example : ( 9474 len (9474)=4, make 4 is power of all input individual number and add for answer ) 9\*\*4 + 4\*\*4 + 7\*\*4 + 4\*\*4 = 9474 the num is an Armstrong**

* **Read the input number.**
* **Len(input\_str) before that convert into string because make iterate. str(num)**
* **Sum of power = 0.**
* **For loop for make string iteration and make len(num) as power of each number**
* **If the sum is equal to the original number, print "Armstrong Number".**
* **Else, print "Not an Armstrong Number".**

### **7. Swapping First and Last Characters of a String**

**Write logic:**

* **Read the input string.**
* **Check the string has at least 2 characters by if len(num)<2.**
* **Swap the first and last characters while keeping the middle part unchanged.**
  + **swap\_fix=swap[-1]+swap[1:-1]+swap[0]**
* **Print the swap\_fix string.**

### 

**8. Converting Decimal to Binary**

**Write logic:**

* **num=int(input("Enter the number"))**
* **binary=""**
* **while num>0:**

1. **rem=num%2**
2. **binary = binary+str(rem)**
3. **num=num//2**

* **binary=binary[::-1]**
* **print ("Binary of num:", binary)**

### **9. Finding the Longest Word in a Sentence**

**Write logic:**

* **sentence=input("Enter the sentence")**
* **words=sentence.split()**
* **longest=""**
* **for word in words:**

1. **if len(word) > len(longest):**
2. **longest=word**

* **print("The longest word is :", longest)**

### **10. Checking if Two Strings are Anagrams**

**Write logic:**

* **str1 = input("Enter the first string: ")**
* **str2 = input("Enter the second string: ")**
* **# Remove spaces and convert to lowercase**
* **str1\_clean = str1.replace(" ", "").lower()**
* **str2\_clean = str2.replace(" ", "").lower()**
* **# Sort the characters of both strings**
* **sorted\_str1 = sorted(str1\_clean)**
* **sorted\_str2 = sorted(str2\_clean)**
* **if sorted\_str1 == sorted\_str2:**

1. **print("Anagram")**
2. **else:**
3. **print("Not an Anagram")**

* **print (sorted\_str1,sorted\_str2)**