# Exercise: Loops

Test your tasks in the Judge system: <https://judge.softuni.org/Contests/4412>

# For Loop

## Power of Number

Write a program to calculate number raising to a certain power:

* Reads an **integer** **number n** from the console
* Reads an **integer number p** which represents **power** from the console
* Print the result of **n to the power of p**

**Note:** Don't use **Math.Pow()**, use loops

ANSWER:

int n = int.Parse(Console.ReadLine());

int p = int.Parse(Console.ReadLine());

// Calculate n to the power of p using a loop

int result = 1;

for (int i = 0; i < p; i++)

{

result \*= n;

}

// Print the result

Console.WriteLine($"{result}");

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  5 | 32 | 3  4 | 81 | 2  3 | 8 |

## Multiplication Table

Write a program that prints a **multiplication table**:

* Reads an **integer number** **n** from the console
* Print a **multiplication table** of size 10 for given integer **n** in the following format:

"**{n} x {i} = {result}**" for each **i** in the range [1…10]

ANSWER:

int n = int.Parse(Console.ReadLine());

// Print the multiplication table for the given number

for (int i = 1; i <= 10; i++)

{

int result = n \* i;

Console.WriteLine($"{n} x {i} = {result}");

}

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2 | **2** **x 1 = 2**  **2 x 2 = 4**  **2 x 3 = 6**  **2 x 4 = 8**  **2 x 5 = 10**  **2 x 6 = 12**  **2 x 7 = 14**  **2 x 8 = 16**  **2** **x 9 = 18**  **2** **x 10 = 20** | 3 | **3** **x 1 = 3**  **3 x 2 = 6**  **3 x 3 = 9**  **3 x 4 = 12**  **3 x 5 = 15**  **3 x 6 = 18**  **3 x 7 = 21**  **3 x 8 = 24**  **3** **x 9 = 27**  **3** **x 10 = 30** | 5 | **5** **x 1 = 5**  **5 x 2 = 10**  **5 x 3 = 15**  **5 x 4 = 20**  **5 x 5 = 25**  **5 x 6 = 30**  **5 x 7 = 35**  **5 x 8 = 40**  **5** **x 9 = 45**  **5** **x 10 = 50** |

## Biggest Number

Write a program to find **the biggest** among given **n** numbers:

* Read an integer number **n** (the **amount** of input numbers) and **n** integer numbers from the console
* Find and print the **biggest** number

### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 3  40  90  50 | 90 | 4  -40  -3  -90  -50 | -3 | 2  1  7 | 7 |

ANSWER:

int n = int.Parse(Console.ReadLine());

// Initialize the largest number with the minimum possible value

int largest = int.MinValue;

// Read the integers and find the largest one

for (int i = 0; i < n; i++)

{

int num = int.Parse(Console.ReadLine());

if (num > largest)

{

largest = num;

}

}

// Print the largest number

Console.WriteLine($"{largest}");

## Vowel Sum

Write a program to **sum N vowels**, according to the table below:

* Read an integer number **N**: the count of characters
* Read **N characters** and for each vowel character adds its value from the table to the result

ANSWER:

int n = int.Parse(Console.ReadLine());

int sum = 0;

// Process each character

for (int i = 0; i < n; i++)

{

char ch = char.Parse(Console.ReadLine().ToLower());

// Add the value based on the vowel table

switch (ch)

{

case 'a':

sum += 1;

break;

case 'e':

sum += 2;

break;

case 'i':

sum += 3;

break;

case 'o':

sum += 4;

break;

case 'u':

sum += 5;

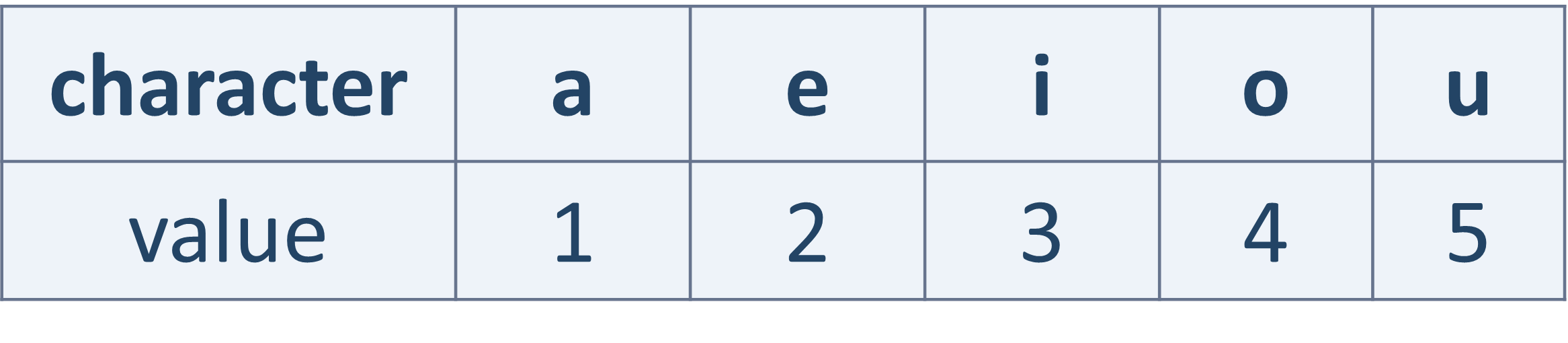
break;

}

}

// Print the result

Console.WriteLine($"{sum}");



### **Example**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  a  e | 3 | 3  i  x  u | 8 | 3  o  g  a | 5 |

## Division to 2, 3 and 4

Write a program to find **statistics about division to 2, 3 and 4**:

* Read an **integer number N** and **N** integers from the console
* Find in **percentages** how many of these integers can divide without a remainder to numbers **2**, **3** and **4**
* Print the percentages, formatted to the **second** decimal digit:
  + On the first line print **percent** of the numbers that are **divisible by 2**
  + On the first line print **percent** of the numbers that are **divisible by 3**
  + On the first line print **percent** of the numbers that are **divisible by 4**
  + ANSWER:
  + int n = int.Parse(Console.ReadLine());
  + // Initialize counters for each divisibility check
  + int divisibleBy2 = 0;
  + int divisibleBy3 = 0;
  + int divisibleBy4 = 0;
  + // Read the integers and count divisibility
  + for (int i = 0; i < n; i++)
  + {
  + int num = int.Parse(Console.ReadLine());
  + if (num % 2 == 0)
  + divisibleBy2++;
  + if (num % 3 == 0)
  + divisibleBy3++;
  + if (num % 4 == 0)
  + divisibleBy4++;
  + }
  + // Calculate percentages
  + double percentDivisibleBy2 = (double)divisibleBy2 / n \* 100;
  + double percentDivisibleBy3 = (double)divisibleBy3 / n \* 100;
  + double percentDivisibleBy4 = (double)divisibleBy4 / n \* 100;
  + // Print the results formatted to the second decimal digit
  + Console.WriteLine($"{percentDivisibleBy2:F2}%");
  + Console.WriteLine($"{percentDivisibleBy3:F2}%");
  + Console.WriteLine($"{percentDivisibleBy4:F2}%");

### **Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 3  3  6  9 | 33.33%  100.00%  0.00% | 3  4  6  3 | 66.67%  66.67%  33.33% |

# While Loop

## Special Number

Write a program to check if given number is **special**:

* + **Special numbers are divisible by all of their digits without remainder**
  + Reads an integer number and check whether it is a special number
  + If the number **IS** special print: "**{num} is special**"
  + If the number is **NOT** special print: "**{num} is not special**"

**Note:** There will not be numbers with digit 0 in them.

ANSWER:

int number = int.Parse(Console.ReadLine());

// Copy of the original number for printing the result

int originalNumber = number;

// Flag to check if the number is special

bool isSpecial = true;

// Extract each digit and check divisibility

while (number > 0)

{

int digit = number % 10;

if (digit == 0 || originalNumber % digit != 0)

{

isSpecial = false;

break;

}

number /= 10;

}

// Print the result

if (isSpecial)

{

Console.WriteLine($"{originalNumber} is special");

}

else

{

Console.WriteLine($"{originalNumber} is not special");

}

### **Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 23 | 23 is not special | 212 | 212 is special |

## Special Bonus

Write a program to apply a **20% bonus** for certain number:

* + Reads **an integer number** from the console: the "**stop number**"
  + Keep **reading integers** until it finds the **stop number**
  + When the **stop number** is found, increase the value of the **previous** number **before it** with **20%** and print it

ANSWER;

int stopNumber = int.Parse(Console.ReadLine());

// Initialize variables

int previousNumber = 0;

bool isFirstNumber = true;

while (true)

{

// Read the next integer from the console

int currentNumber = int.Parse(Console.ReadLine());

if (currentNumber == stopNumber)

{

if (isFirstNumber)

{

Console.WriteLine("No previous number to apply a bonus.");

}

else

{

// Apply 20% bonus to the previous number

double bonus = previousNumber \* 0.20;

double result = previousNumber + bonus;

Console.WriteLine($"{result}");

}

break;

}

// Update the previous number and set the first number flag to false

previousNumber = currentNumber;

isFirstNumber = false;

}

### **Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 25  20  30  25 | 36 | 14  6  10  14 | 12 |

## Account Balance

Write a program to calculate an **account balance**:

* + Read a sequence of **incomes** / **expenses**, until "**End**" is read
  + Add the money to the balance (starting form 0)
  + Print "**Increase: {money}**" or "**Decrease: {money}**", where value is formatted to the **second** decimal digit
  + Finally, print the **account balance,** formatted to the **second** decimal digit in the following format: "**Balance: {account balance}**"

ANSWER:

double balance = 0.0;

while (true)

{

// Read the next input from the console

string input = Console.ReadLine();

// Check for the end condition

if (input == "End")

{

break;

}

// Convert input to double

double money;

if (double.TryParse(input, out money))

{

// Update balance and print the corresponding message

if (money > 0)

{

balance += money;

Console.WriteLine($"Increase: {money:F2}");

}

else

{

balance += money; // Adding a negative number (expense)

Console.WriteLine($"Decrease: {Math.Abs(money):F2}");

}

}

else

{

Console.WriteLine("Invalid input. Please enter a valid number or 'End' to finish.");

}

}

// Print the final balance

Console.WriteLine($"Balance: {balance:F2}");

### **Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 500  15.5  -80.35  End | Increase: 500.00  Increase: 15.50  Decrease: 80.35  Balance: 435.15 | 200  300  -100  End | Increase: 200.00  Increase: 300.00  Decrease: 100.00  Balance: 400.00 |