puzzles based on **decision-making, branching, and looping**:

---

1. FizzBuzz

Print numbers from 1 to 100. For multiples of 3, print "Fizz" instead of the number, for multiples of 5, print "Buzz," and for multiples of both, print "FizzBuzz."

Example Output:
```
1, 2, Fizz, 4, Buzz, Fizz, 7, ...
```

```
#include <stdio.h>

int main() {
   for (int i = 1; i <= 100; i++) {
      if (i % 3 == 0 && i % 5 == 0) {
         printf("FizzBuzz ");
      } else if (i % 3 == 0) {
         printf("Fizz ");
      } else if (i % 5 == 0) {
         printf("Buzz ");
      } else {
         printf("%
```

---

2. Prime Numbers

Write a program to print all prime numbers between 1 and `n`.

Input: `n = 20`
Output: `2, 3, 5, 7, 11, 13, 17, 19`

```
#include <stdio.h>
#include <stdbool.h>

void printPrimes(int n) {
   for (int num = 2; num <= n; num++) {
      bool isPrime = true;
      for (int i = 2; i * i <= num; i++) {
         if (num % i == 0) {
            isPrime = false;
            break;
         }
      }
      if (isPrime) {
```

```
            printf("%d ", num);
        }
    }
    printf("\n");
}

int main() {
    int n = 20;
    printf("Prime numbers up to %d: ", n);
    printPrimes(n);
    return 0;
}
```

---

3. Reverse a Number

Write a program to reverse a given number using a loop.

Input: `n = 12345`
Output: `54321`

```
#include <stdio.h>

int reverseNumber(int n) {
    int reversed = 0;
    while (n != 0) {
        reversed = reversed * 10 + n % 10;
        n /= 10;
    }
    return reversed;
}

int main() {
    int num = 12345;
    printf("Reversed Number: %d\n", reverseNumber(num));
    return 0;
}
```

---

4. Check Armstrong Number

An Armstrong number is a number that is equal to the sum of its own digits raised to the power of the number of digits. Write a program to check if a number is an Armstrong number.

Input: `n = 153`
Output: `Yes` ($153 = 1^3 + 5^3 + 3^3$)

```c
#include <stdio.h>
#include <math.h>

int isArmstrong(int n) {
    int original = n, sum = 0, digits = 0;
    while (n > 0) {
        n /= 10;
        digits++;
    }
    n = original;
    while (n > 0) {
        int digit = n % 10;
        sum += pow(digit, digits);
        n /= 10;
    }
    return sum == original;
}

int main() {
    int num = 153;
    if (isArmstrong(num)) {
        printf("%d is an Armstrong number.\n", num);
    } else {
        printf("%d is not an Armstrong number.\n", num);
    }
    return 0;
}
```

---

5. Find Factorial

Write a program to calculate the factorial of a number using a loop.

Input: `n = 5`
Output: `120` (5! = 5 × 4 × 3 × 2 × 1)

```c
#include <stdio.h>

int factorial(int n) {
    int fact = 1;
    for (int i = 1; i <= n; i++) {
        fact *= i;
    }
    return fact;
}
```

```
int main() {
   int num = 5;
   printf("Factorial of %d is %d\n", num, factorial(num));
   return 0;
}
```

---

6. Number Pattern

Print the following pattern using loops:

```
1
12
123
1234
12345
```

```
#include <stdio.h>

int main() {
   int n = 5;
   for (int i = 1; i <= n; i++) {
      for (int j = 1; j <= i; j++) {
         printf("%d", j);
      }
      printf("\n");
   }
   return 0;
}
```

---

7. Sum of Digits

Write a program to calculate the sum of digits of a number.

Input: `n = 1234`
Output: `10` (1 + 2 + 3 + 4)

```
#include <stdio.h>

int sumOfDigits(int n) {
   int sum = 0;
   while (n != 0) {
      sum += n % 10;
      n /= 10;
   }
   return sum;
}
```

```
int main() {
    int num = 1234;
    printf("Sum of digits: %d\n", sumOfDigits(num));
    return 0;
}
```

---

8. Generate Fibonacci Sequence

Write a program to generate the first `n` terms of the Fibonacci sequence.

Input: `n = 10`
Output: `0, 1, 1, 2, 3, 5, 8, 13, 21, 34`

```
#include <stdio.h>

void generateFibonacci(int n) {
    int a = 0, b = 1;
    printf("%d %d ", a, b);
    for (int i = 3; i <= n; i++) {
        int c = a + b;
        printf("%d ", c);
        a = b;
        b = c;
    }
    printf("\n");
}

int main() {
    int n = 10;
    printf("Fibonacci sequence up to %d terms: ", n);
    generateFibonacci(n);
    return 0;
}
```

---

9. Count Vowels and Consonants

Write a program to count the number of vowels and consonants in a given string.

Input: `"hello"`
Output: `Vowels: 2, Consonants: 3`

```
#include <stdio.h>
#include <ctype.h>
```

```
void countVowelsConsonants(char str[]) {
   int vowels = 0, consonants = 0;

   for (int i = 0; str[i] != '\0'; i++) {
      char ch = tolower(str[i]);
      if (ch >= 'a' && ch <= 'z') {
         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
            vowels++;
         } else {
            consonants++;
         }
      }
   }

   printf("Vowels: %d, Consonants: %d\n", vowels, consonants);
}

int main() {
   char str[] = "hello";
   countVowelsConsonants(str);
   return 0;
}
```

---

10. Guess the Number Game

Write a program to implement a "Guess the Number" game. The program randomly selects a number, and the user tries to guess it, receiving hints of "higher" or "lower."

Example Interaction:

- Program: Guess the number (1–100):
- User: `50`
- Program: Higher
- User: `75`
- Program: Lower
- User: `63`
- Program: Correct!

```
#include <stdio.h>

#include <stdlib.h>

#include <time.h>
```

```
int main() {
    srand(time(0));
    int target = rand() % 100 + 1;  // Random number between 1 and 100
    int guess, attempts = 0;

    printf("Guess the number (1-100): ");
    while (1) {
        scanf("%d", &guess);
        attempts++;
        if (guess < target) {
            printf("Higher\n");
        } else if (guess > target) {
            printf("Lower\n");
        } else {
            printf("Correct! You guessed it in %d attempts.\n", attempts);
            break;
        }
    }

    return 0;
}
```