Bugs In Code

Find bugs in this code:

```
#include <stdio.h>

void decimalToBinary(int n) {
    int binary[100], i = 0;

    while (n != 0) {
        binary[i] = n % 2;
        n /= 2;
        i++;
    }

    printf("Binary: ");
    for (int j = i - 1; j >= 0; j--) {
        printf("%d", binary[j]);
    }
    printf("\n");
}

void decimalToOctal(int n) {
    int octal[100], i = 0;

    while (n != 0) {
        octal[i] = n % 8;
        n /= 8;
        i++;
    }

    printf("Octal: ");
    for (int j = i - 1; j >= 0; j--) {
        printf("%d", octal[j]);
    }
    printf("\n");
}

void decimalToHexadecimal(int n) {
    int hexadecimal[100], i = 0;

    while (n != 0) {
        hexadecimal[i] = n % 16;
```

```
        n /= 16;
        i++;
    }

    printf("Hexadecimal: ");
    for (int j = i - 1; j >= 0; j--) {
        if (hexadecimal[j] < 10) {
            printf("%d", hexadecimal[j]);
        } else {
            printf("%c", hexadecimal[j] + 55);
        }
    }
    printf("\n");
}

int main() {
    int decimal;

    printf("Enter a decimal number: ");
    scanf("%d", &decimal);

    decimalToBinary(decimal);
    decimalToOctal(decimal);
    decimalToHexadecimal(decimal);

    return 0;
}
```

[redacted]

[redacted]

[redacted]

Find bugs in this:

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

int main() {
```

```
    char input[100];
    int i;

    printf("Enter a string: ");
    scanf("%s", input);

    for (i = 0; input[i] != '\0'; i++) {
        if (isalpha(input[i]) == 1) {
            printf("%c", input[i]);
        }
    }
    printf("\n");

    return 0;
}
```

Fill in the Blank

```
#include <stdio.h>

struct _____________ {
    _______________;
    _____;
    __________;
};

int main() {
   struct _______________;

    // Fill in the blanks to input student information
    printf("Enter student name: ");
    scanf("%s", _____);
    printf("Enter student age: ");
    scanf("%d", _____);
    printf("Enter student grade: ");
    scanf(" %c", _____);

    // Fill in the blanks to output student information
    printf("\nStudent Information:\n");
    printf("Name: %s\n", _____);
    printf("Age: %d\n", _____);
    printf("Grade: %c\n", _____);

    return 0;
}
```

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

```
#include <stdio.h>

int _________ (______) {
    // Base case: factorial of 0 is 1
    if (n == ______) {
        return ______;
    }
    // Recursive case: calculate factorial
    else {
        return n * factorial(______);
    }
}

int main() {
    int num, fact;

    // Fill in the blanks to input a number
    printf("Enter a positive integer: ");
    scanf("%d", ______);

    // Fill in the blanks to call the factorial function
    fact = _________(__________);

    // Fill in the blanks to output the factorial
    printf("Factorial of %d = %d\n", ______, ______);

    return 0;
}
```

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

```
#include <stdio.h>

int main() {
    int num, i;
    float sum = 0;

    // Fill in the blanks to input the number of terms
    printf("Enter the number of terms: ");
    scanf("%d", _____);

    for (i = _______; _____; i_____) {
        // Fill in the blanks to calculate the sum of the series
        sum += __________;
    }
```

```
    // Fill in the blanks to output the sum of the series
    printf("Sum of the series = %.2f\n", _____);

    return 0;
}
```

[redacted]

Determine the Output

1. Given this code, determine the output for these 3 inputs:
    a. 12 30
    b. 1 10
    c. -11 -1

```
#include <stdio.h>
int checkPrimeNumber(int n);
int main() {

  int n1, n2, i, flag;

  printf("Enter two positive integers: ");
  scanf("%d %d", &n1, &n2);
```

```
  // swap n1 and n2 if n1 > n2
  if (n1 > n2) {
    n1 = n1 + n2;
    n2 = n1 - n2;
    n1 = n1 - n2;
  }

  printf("Prime numbers between %d and %d are: ", n1, n2);
  for (i = n1 + 1; i < n2; ++i) {

    // flag will be equal to 1 if i is prime
    flag = checkPrimeNumber(i);

    if (flag == 1) {
      printf("%d ", i);
    }
  }

  return 0;
}

// user-defined function to check prime number
int checkPrimeNumber(int n) {
  int j, flag = 1;

  for (j = 2; j <= n / 2; ++j) {

    if (n % j == 0) {
      flag = 0;
      break;
    }
  }

  return flag;
}
```

[redacted]

Conversions

1. For-While Loop:

```
#include <stdio.h>

int main() {
    int i;

    printf("Even numbers from 1 to 10: ");
    for (i = 1; i <= 10; i++) {
        if (i % 2 == 0) {
            printf("%d ", i);
        }
    }
    printf("\n");

    return 0;
}
```

[redacted]

2. DoWhile - For Loop:

```
#include <stdio.h>

int main() {
    int n, i = 0, a = 0, b = 1, nextTerm;

    printf("Enter the number of terms: ");
    scanf("%d", &n);
```

```
    printf("Fibonacci Series: ");
    do {
        printf("%d, ", a);
        nextTerm = a + b;
        a = b;
        b = nextTerm;
        i++;
    } while (i < n);
    printf("\n");

    return 0;
}
```

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

3. While - DoWhile Loop:

```
#include <stdio.h>

int main() {
    int n, factorial = 1, i = 1;

    printf("Enter a positive integer: ");
    scanf("%d", &n);

    while (i <= n) {
        factorial *= i;
        i++;
    }
```

```
    printf("Factorial of %d = %d\n", n, factorial);

    return 0;
}
```
[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

Determine the Value

```
#include <stdio.h>

int main() {
    int x = 5,
        y = 127;
    char z = 'B';
    float w = 6.28;

    float r1 = w * w; // 39.38
    int r2 = y / 5; // 25
    char r3 = z + 'd' - 'D'; // 'g'
    float r4 = w % (w / 2); // 0.14
    unsigned r5 = y % (x * x); // 2
    float r6 = (int)w + x; // 11
    float r7 = w - (int)w; // 0.28
    unsigned r8 = (x < ++x) ? x + 1 : x - 1; // 7
    int r9 = y == y++; // 1

    return 0;
```

}

[redacted]

CLONEWARS

There are 2 versions (ask em which one they prefer):

1. Easy Version:

```
#include <stdio.h>

int REPUBLIC (int x);
int SEPARATIST (int y);
void NEUTRAL (int z);

int main() {
   int CLONEWARS = 303;
   printf("CLONEWARS = %d\n", CLONEWARS);

   CLONEWARS = REPUBLIC(CLONEWARS);
   printf("CLONEWARS = %d\n", CLONEWARS);

   CLONEWARS += 4; // Simple addition
   printf("CLONEWARS = %d\n", CLONEWARS);

   NEUTRAL(CLONEWARS);
   printf("CLONEWARS = %d\n", CLONEWARS);

   CLONEWARS += 43; // Simple addition
   printf("CLONEWARS = %d\n", CLONEWARS);

   CLONEWARS = SEPARATIST(CLONEWARS);
   printf("CLONEWARS = %d\n", CLONEWARS);

   return 0;
}
```

```
int REPUBLIC (int x) {
    return x + 1; // Simple addition
}
void NEUTRAL (int z) {
    // Does nothing in the super easy version
}
int SEPARATIST (int y) {
    return 150; // Always returns 150 in the super easy version
}
```

[redacted]

2. Hard Version:

```
#include <stdio.h>

int REPUBLIC (int x);
int SEPARATIST (int y);
void NEUTRAL (int z);

int main() {
    int CLONEWARS = 303;
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS = REPUBLIC(CLONEWARS);
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS += ((10 & 7) >> 1) | (8 << 1); // Bitwise operations and value shifting
    printf("CLONEWARS = %d\n", CLONEWARS);

    NEUTRAL(CLONEWARS);
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS += 0x2B; // Simple addition
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS = SEPARATIST(CLONEWARS);
    printf("CLONEWARS = %d\n", CLONEWARS);

    return 0;
```

```
}
int REPUBLIC (int x) {
    return x + 1; // Simple addition
}
void NEUTRAL (int z) {
    // Does nothing in the super hard version
}
int SEPARATIST (int y) {
    return ((y & 0xFF) << 1) | (0x96 >> 1); // Bitwise operations and value shifting
}
```

[redacted]

ASCII Value Questions

01000110 01001111 01010010 00100000 01010100 01001000 01000101 00100000 01010010 01000101 01010000 01010101 01000010 01001100 01001001 01000011 00100001 00100001 00100001 00100001

Convert to all of its counterparts, then give me the sentence. (If you want to do a meme, feel free to stand up and shout it)

[redacted]

MCQ

Q1. Which of the following is an important requirement of c programming?

1. Function
2. Input variables
3. Output variables
4. All of the above

[redacted]

Q2. Which of the following is true in the case of c programming?

1. The function is a variable.
2. Parenthesis isn’t needed at all in c programming.
3. There is no need for closing parenthesis in c programming
4. Float is a variable.

[redacted]

Q3. When was C programming developed?

1. The 1950s
2. The 1960s
3. The 1980s
4. The 1970s

[redacted]

Q4. What was C programming adapted from?

1. C++
2. Combined programming language

3. python
4. All of the above

[redacted]

Q5. Which of the following is not a variable type in c programming?

1. Float
2. Int
3. While loop
4. All of the above

[redacted]

Q6. What is the use of print f in c programming?

1. Helps in the printing of a string on the output screen
2. Processes the variables in a program
3. Is a variable type
4. All of the above

[redacted]

Q7. What are strings in C programming?

1. Individual variables
2. Group of function
3. Group of character type variables in array form
4. All of the above

[redacted]

Q8. What does a do-while loop do?

1. Repeats the process infinitely
2. Processes the code at least once and then repeats
3. Repeats only once
4. All of the above

[redacted]

Q9. What is a while loop?

1. Repeats the loop if the condition applies true
2. Processes the code at least once and then repeats
3. Repeats only once
4. All of the above

[redacted]

Q10. What of the following is true?

1. Variables are functions
2. Variable is a type of output command
3. Variables are used to store values
4. All of the above

[redacted]

Q11. What are float variables?

1. Integer value
2. unknown value
3. Decimal value
4. All of the above

[redacted]

Q12. What is a function?

1. Looping code
2. Code that operates when called
3. Unknown variable
4. All of the above

[redacted]

Q13. How many variables can the following string contain bat[45]?

1. 20
2. 40
3. 44
4. 45

[redacted]

Q14. What is function overloading?

1. Process of multiple functions
2. Multiple functions with the same name

3. Looping functions
4. All of the above

[redacted]

Q15. Which header file uses gets()?

1. Studio. h
2. Stdlib.h
3. Conio.h
4. All of the above

[redacted]

Q16. Which of the following is the wrong way of writing c language?

1. Int bat;
2. Float cat_a;
3. Int @rat
4. All of the above

[redacted]

Q17. What are const data types used for?

1. Unknown values
2. Static or constant values
3. Dynamic variable values
4. All of the above

[redacted]

Q18. What are the primary iterations in C programming?

1. While loop
2. Do while loop
3. For loop
4. All of the above

[redacted]

Q19. Which of the following is not related to c programming?

1. conio.h
2. getch()
3. console.log
4. All of the above

[redacted]

Q20. What is scanf() in c programming?

1. The layout of an input string
2. Array
3. Output function
4. All of the above

[redacted]