

23CST101 – PROBLEM SOLVING AND C PROGRAMMING

PUZZLES

Puzzle 1: Reverse the Output

What will be the output of this code snippet, and why?

```
#include <stdio.h>

struct example {
    char a;
    int b;
};

int main() {
    printf("%lu\n", sizeof(struct example));
    return 0;
}
```

Hint: This tests sequence points and undefined behavior.

Puzzle 2: Infinite or Not?

Does the following program result in an infinite loop? If yes, why? If no, explain the output.

```
#include <stdio.h>

int main() {
    for (unsigned int i = 10; i >= 0; i--) {
        printf("%u\n", i);
    }
    return 0;
}
```

Hint: Think about how unsigned integers behave when they underflow.

Puzzle 3: Find the Error

The following program compiles successfully but crashes at runtime. Why?

```
#include <stdio.h>

int main() {

    int *ptr = NULL;

    *ptr = 42;

    printf("%d\n", *ptr);

    return 0;

}
```

Hint: Understand what happens when dereferencing a `NULL` pointer.

Puzzle 4: Array vs Pointer

What is the output of the following program? Explain your reasoning.

```
#include <stdio.h>

int main() {

    int arr[] = {10, 20, 30, 40};

    int *ptr = arr;

    printf("%d %d %d\n", *ptr, *(ptr + 1), ptr[2]);

    return 0;

}
```

Hint: Explore how arrays and pointers are interconnected in C.

Puzzle 5: Function Behavior

What will the following code output? Why?

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

```

void change(int *ptr) {
    int x = 10;
    ptr = &x;
    *ptr = 20;
}

int main() {
    int a = 5;
    int *p = &a;
    change(p);
    printf("%d\n", a);
    return 0;
}

```

Hint: Think about whether the pointer passed to the function is modified or not.

Puzzle 6: Odd Behavior

What will this program print?

```

#include <stdio.h>

int main() {
    int x = 5;
    if (x == 5 && printf("Hello ")) {
        printf("World!\n");
    }
    return 0;
}

```

Hint: Consider how `printf` works in conditions and the evaluation order of `&&`.

Puzzle 7: Size of Structures

What will the following program output, and why?

```
#include <stdio.h>

struct example {
    char a;

    int b;
};

int main() {
    printf("%lu\n", sizeof(struct example));

    return 0;
}
```

Hint: Padding and alignment rules.