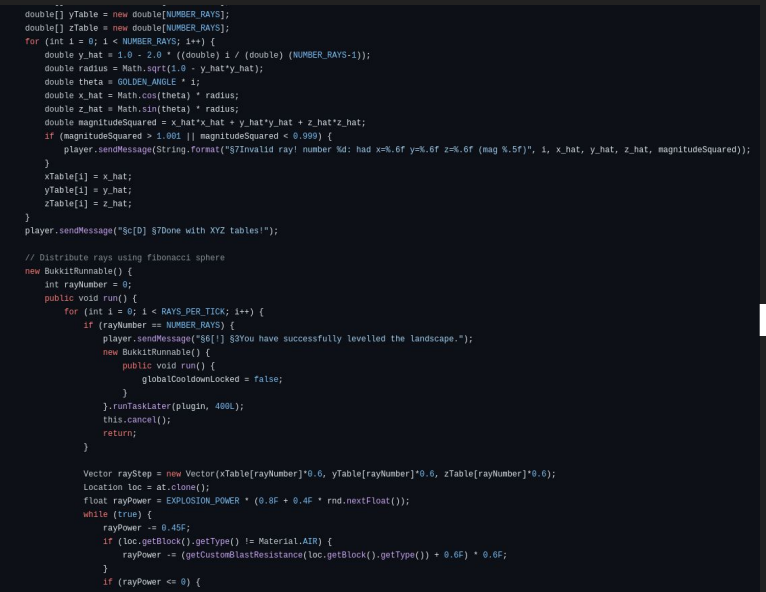
Capture The Flag

WL Hack Club

intro

what is a ctf

- Reverse engineering challenge
- Extract the flag from the source code



```
double[] yTable = new double[NUMBER_RAYS];
double[] zTable = new double[NUMBER_RAYS];
for (int i = 0; i < NUMBER_RAYS; i++) {
    double y_hat = 1.0 - 2.0 * ((double) i / (double) (NUMBER_RAYS-1));
    double radius = Math.sqrt(1.0 - y_hat*y_hat);
    double theta = GOLDEN_ANGLE * i;
    double x_hat = Math.cos(theta) * radius;
    double z_hat = Math.sin(theta) * radius;
    double magnitudeSquared = x_hat*x_hat + y_hat*y_hat + z_hat*z_hat;
    if (magnitudeSquared > 1.001 || magnitudeSquared < 0.999) {
        player.sendMessage(String.format("§7Invalid ray! number %d: had x=%.6f y=%.6f z=%.6f (mag %.5f)", i, x_hat, y_hat, z_hat, magnitudeSquared));
    }
    xTable[i] = x_hat;
    yTable[i] = y_hat;
    zTable[i] = z_hat;
}
player.sendMessage("§c[D] §7Done with XYZ tables!");

// Distribute rays using fibonacci sphere
new BukkitRunnable() {
    int rayNumber = 0;
    public void run() {
        for (int i = 0; i < RAYS_PER_TICK; i++) {
            if (rayNumber == NUMBER_RAYS) {
                player.sendMessage("§6[!] §3You have successfully levelled the landscape.");
                new BukkitRunnable() {
                    public void run() {
                        globalCooldownLocked = false;
                    }
                }.runTaskLater(plugin, 400L);
                this.cancel();
                return;
            }

            Vector rayStep = new Vector(xTable[rayNumber]*0.6, yTable[rayNumber]*0.6, zTable[rayNumber]*0.6);
            Location loc = at.clone();
            float rayPower = EXPLOSION_POWER * (0.8F + 0.4F * rnd.nextFloat());
            while (true) {
                rayPower -= 0.45F;
                if (loc.getBlock().getType() != Material.AIR) {
                    rayPower -= (getCustomBlastResistance(loc.getBlock().getType()) + 0.6F) * 0.6F;
                }
                if (rayPower <= 0) {
```

flag=123456789

intro

setup

- Involves both static and dynamic analysis
- You probably want a place to work
- **Open a new python window in repl.it**

intro

warmup

```
f = int(input("flag:"))
if f - 500 > 400 and f + 100 < 1002:
    print("success")
else:
    print("failure")
```

intro

warmup solution

```
f = int(input("flag:"))
if f - 500 > 400 and f + 100 < 1002:
    print("success")
else:
    print("failure")
```

The only value of flag that satisfies this condition is 901.

ARE YOU READY?

I

capture the flag

Code:
https://raw.githubusercontent.com/WLHackClub/ctf/main/level1.py

Flag Finding Strategies:

- Analyze
- Mess around
- Guess & check

I

analysis

```
f = input("Enter password: ")
a = int(f[0])
b = int(f[1])
c = int(f[2])
d = int(f[3])
if b != c:
    print('ACCESS DENIED')
elif c != d:
    print('ACCESS DENIED')
```

a, b, c, d are the 4 digits of the password

If b isn't equal to c, or if c isn't equal to d, it denies us. So b, c, and d must be equal.

I

analysis

```
elif a != b - 2:
    print('ACCESS DENIED')
elif a + b + c == 25:
    print('ACCESS GRANTED')
else:
    print('ACCESS DENIED')
```

a must equal b-2

a+b+c must equal 25

time for some math…

II

ARE YOU READY?

II

capture the flag

Code:
https://raw.githubusercontent.com/WLHackClub/ctf/main/level2.py

This one requires you to know `bin()`.

Try to see if you can tell what it does!

II

hint

The `bin()` function converts a number to binary, in this format:

```
bin(4) = “0b100”

bin(6) = “0b110”

bin(32) = “0b100000”

bin(69) = “0b1000101”
```

II

analysis

```
flag = int(input("Enter password: "))
c = bin(flag)
if len(c) != 10:
    print("ACCESS DENIED")
elif c[2] != "1":
    print("ACCESS DENIED")
elif c[3] != "0":
    print("ACCESS DENIED")
elif c[4:9] != "00000":
    print("ACCESS DENIED")
elif c[9] != "1":
    print("ACCESS DENIED")
else:
    print("ACCESS GRANTED")
```

c is in binary

c has a length of 10: 0bXXXXYYYY

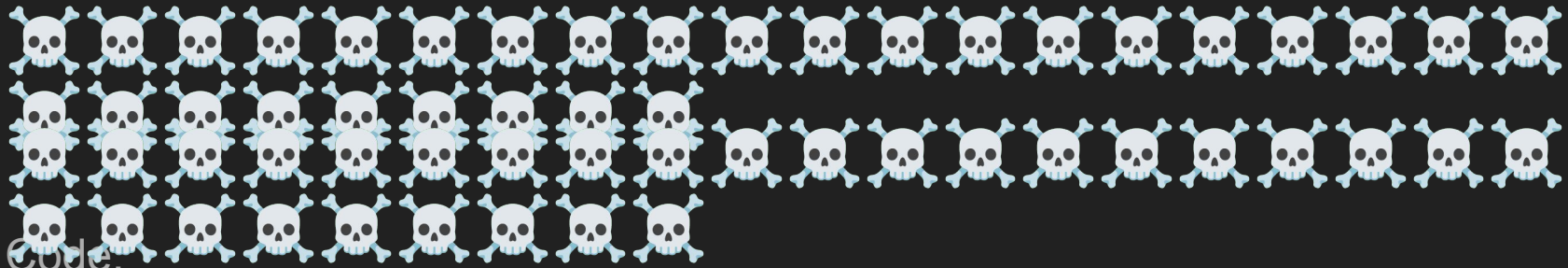
0b1XXXYYYY

0b10XXYYYY

0b1000000Y

0b10000001 = 129

ARE YOU READY?



Code:

https://raw.githubusercontent.com/WLHackClub/ctf/main/level3.py

Your best first step is to try to simplify the code.


NO CHEATING

This CTF uses control flow flattening.

As you can probably tell, this makes it extremely difficult to understand what's going on.

But we can un-flatten our code - and we need to do this to get the flag.

```
i = 0
while True:
    if i == 1: ...
    if i == 2: ...
    if i == 3: ...
    if i == 4: ...
```

☠️

☠️

```
i = 4
while True:
    ...
    elif i == 2:
        a = int(flag) % 100
        b = int(flag) // 100
        i = 6
    ...
    elif i == 4:
        flag = input("Enter
password: ")
        i = 2
```

```
i = 4
while True:
    ...
    elif i == 4:
        flag = input("Enter
password: ")
        a = int(flag) % 100
        b = int(flag) // 100
        i = 6
```

☠️

☠️

```
    ...
    elif i == 4:
        flag = input("Enter
password: ")
        a = int(flag) % 100
        b = int(flag) // 100
        i = 6
    ...
    elif i == 6:
        if b > 100:
            print("ACCESS DENIED")
            break
        i = 1
```

```
    ...
    elif i == 4:
        flag = input("Enter
password: ")
        a = int(flag) % 100
        b = int(flag) // 100
        if b > 100:
            print("ACCESS DENIED")
            break
        i = 1
```

Repeating this process, we get our final version of the code.
Notice how there were fake branches that were made to mislead you.

```
elif i == 5:
    if flag == "1447":
        print("ACCESS GRANTED")
```

```
flag = input("Enter password: ")
a = int(flag) % 100
b = int(flag) // 100
if b > 100:
    print("ACCESS DENIED")
    break
if a != b:
    print("ACCESS DENIED")
    break
if a > 20:
    print("ACCESS DENIED")
    break
if a < 10:
    print("ACCESS DENIED")
    break
if a % 10 != 4:
    print("ACCESS DENIED")
    break
print("ACCESS GRANTED")
break
```