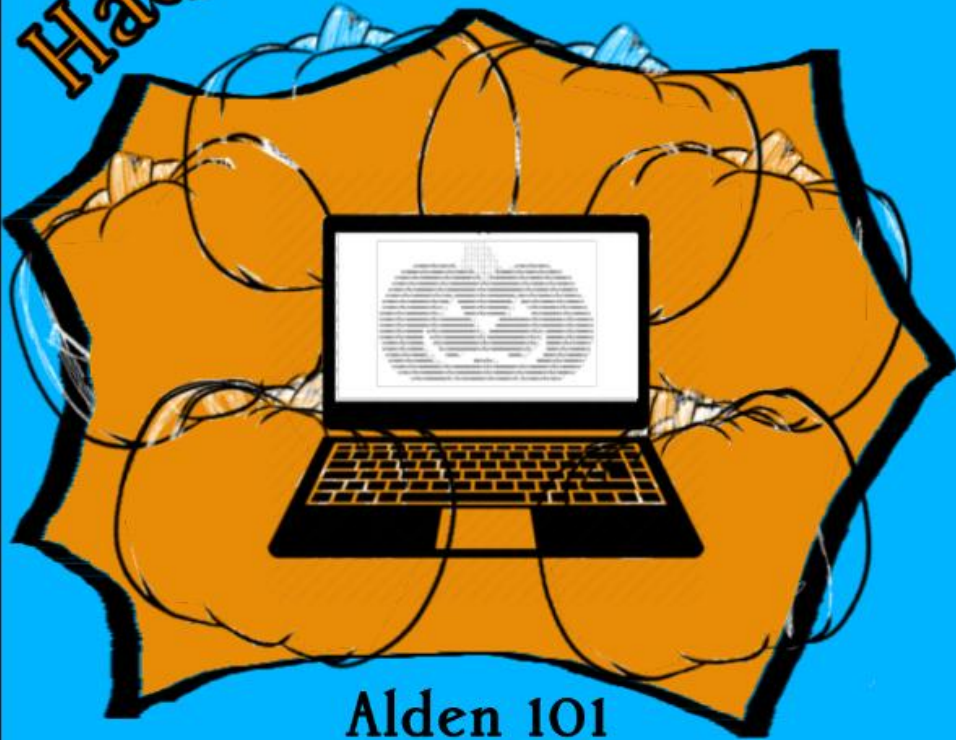


CMPSC 100

Computational Expression

Association for Computing Machinery

Hacktoberfest



Alden 101

Wednesday October 23rd 5:00pm - 10:00pm

Email us at acm@allegheny.edu
for the RSVP form

Participate in a (not so ancient)
October tradition...

(no, not the World Series)

HACKTOBERFEST

Wednesday, 23 October

Alden 101

5:00 - 10:00 PM

Review sessions

- Based on feedback, I will hold the review session on:

FRIDAY, 25 OCTOBER 12:00PM – 1:00PM, ALDEN 109

- The review repository (with guidelines) is posted to GitHub, the link is in the #general tab in our Slack.
 - You do not need to attend the review session to complete the assignment, **but it is highly suggested**.
 - If the above review time does not work for you, please schedule office hours.
- See me during office hours for any questions you have.

Statements

`int a = 5;`  assignment

`System.out.println("Hello, World!");`  method call

`if (a < 6) {`

`a++;`  "if" statement (conditional)

`}`


while statements

- Create a “loop” in a program’s `flow-of-control`.
- Test a condition over and over to determine whether or not to continue.
 - This test is a `boolean` test meaning conditions are `true/false`.
- Each “loop” is known as an `iteration`.
- The loop keeps doing work until the given condition is met.

while statements

- Appears in the form

```
while (condition) {  
    /*  
        * Tasks to “iterate”  
        */  
}
```



boolean condition

```
int count = 9;
while (count > 0) {
    System.out.println(count + 1);
    count--;
}
System.out.println("LIFTOFF")
```

The "decrement"

10

9

8

7

6

5

4

3

2

1

LIFTOFF




```
int count = 0;
int shift = 4;
String originalWord = "Caesar!";
String encipheredWord new String();
int wordLen = originalWord.length()
while (count < wordLen - 1) {
    encipheredWord += (char)originalWord.charAt(count) + 4;
    count++;
}
System.out.println(originalWorld + "\n" + encipheredWord);
```

Caesar!


Geiwev%

Odd and Even number sorter

We need to track the number currently being tested



Given the set of numbers 1...100:

- Print “Odd” for every odd number.
 - Print “Even” for each even number.
- 




We need some kind of test for this...what might that be?

Update class-activities repository

In the main folder of *your* class-activities repository:




```
git pull download master
```

```
cd 21-October/in-class
```

```
public static void main(String[] args) {  
    int number = 1;  Tracks our numbers  
    while(number <= 100){  
        if(number % 2 == 0) {  Test for even numbers  
            System.out.println(number + ": even");  
        } else {  
            System.out.println(number + ": odd");  
        }  
        number++;  Increases number by 1  
    }  
}
```

The "increment"



```
public static void main(String[] args) {  
    int number = 1;  Tracks our numbers  
    while(number <= 100){  
        System.out.print(number + ": ");  
        if(number % 2 == 0) {  Test for even numbers  
            System.out.println("even");  
        } else {  
            System.out.println("odd");  
        }  
        number++;  Increases number by 1  
    }  
}
```

Testing

Use the `gradle run` command to test

Teaching and old computer old tricks

In a world where there's a computer which doesn't understand the $+$ or $-$ operator, somehow we need to add and subtract numbers...

Without either operator how can we do it?

Teaching and old computer old tricks

- In this problem, `1 + 2` or `3 - 4` doesn't work intuitively.
- We have:
 - The increment `++`
 - The decrement `--`
- Somehow we have to cobble together how adding and subtracting can work in terms of `while` loops.
- There's a solution in the `after-class` folder in today's `class-activities` content.

ADDING

```
while (b != 0) {  
    a++;  
    b--;  
}
```

SUBTRACTING

```
while (b != 0) {  
    a--;  
    b--;  
}
```

In both cases, a is the result.


CODE FROM MAIN METHOD USING SUM

```
Sum sum = new Sum(a,b);  
System.out.println("The sum is " + sum);
```

JSS pg. 152-154

CODE IMPLEMENTING SUM (Sum.java)

```
public class Sum {  
  
    private final int a;  
  
    public Sum(int a, int b) {  
        while (b != 0) {  
            a++;  
            b--;  
        }  
        this.a = a;  
    }  
  
    public String toString(){  
        return Integer.toString(this.a);  
    }  
}
```



after-class folder

Test the content of the `after-class` folder using:

```
gradle -q --console plain run
```