**Name:**

**Date:**

**Period:**

**Parity Tasksheet**

**Part 1:**

**Binary (Base 2) Conversion Practice**

*Convert the following Base 10 values into Binary (Base 2).*

a. 9 b. 45 c. 20

d. 12 e. 4 f. 15

*Choose a time of day. (For example I will choose 5:30pm)*

*Your chosen time of day:*  \_\_\_ **:** \_\_\_\_\_\_ \_\_\_

Let’s convert your chosen time of day into a binary code. There will be eleven binary digits (called “bits”) in your code.

Let the very first bit represent am or pm. Let it be zero for am, and let it be one for pm.

Let the next four digits represent the hour in binary.

Let the next six digits represent the minute in binary.

Example (5:30pm):

1 0101 011110

pm 5 30

*Your time of day in binary code:*

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

*We’ll pretend you are sending this time of day binary string to a friend. To make sure the string is not corrupted, we’ll learn how to protect this binary code you’ve made with a Hamming Code.*

**Part 2:**

**Parity, Parity Bits, and Parity Checks**

“Parity” comes from a latin root “par” which means “equal”. You have probably heard of *income disparity*, which is the quality of two income classes being *unequal*. Maybe you have heard of *par* in golfing, which means taking a number of swings equal to the expected number of swings. The latin “par” also is a root of the word “pair”, as in *a group of two*.

What other words can you think of that have the root “par”?

*pareja* (spanish) -- a (romantic) couple

In mathematics, “parity” is the quality of a number of being either even or odd.

*Examples:*

The number 3 has an odd parity.

The number 12 has an even parity.

The number 5 and 9 have the same parity.

Now consider a binary string, like the one below. A binary string’s parity is even if the string has an even number of “1”s. A binary string’s parity is odd if the string has an odd number of “1”s.

10101011110

What is the parity of the above binary string? Well, how many ones are there? There are seven ones, so the parity of the string is odd.

What are the parities of the following strings?

a. 01001111010 b. 10011110011

Notice that if we change just one of the bits, then the parity will change from odd to even or from even to odd. Maybe we can use this fact to detect whether or not there was an error in the message transmission. But how?

Notice that each binary string so far is exactly eleven bits long. Let’s add one more bit, on the end, and we’ll call it the *Parity Bit*. To tell the parity bit apart from the message bits, we will color the bit red. We will choose the value of this bit to make sure the overall parity is always even.

*Example:*

10101011110 10101011110**1**

No parity bit -- Odd Parity With Parity Bit -- Even Parity

Attach a parity bit to the following binary strings. Ensure that the resulting parity is always even.

a. 01001111010 b. 10011110011

Copy down your chosen time of day binary code from part (a). Attach to it an appropriate parity bit to ensure the overall parity is even.

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

Observe the strings below. Each has a parity bit already attached. Check whether the overall parity of each string is even or not. If the parity is not even, then you can know that the string was corrupted in transmission. This is called performing a “Parity Check”.

a. 101110010110 b. 000010110001

Parity: Parity:

Was there an error? Was there an error?

Because this new parity bit ensures that the parity of any message should be even, we can know that any message that arrives with an odd parity is guaranteed to have an error somewhere.

This parity bit has a special job. If we add more bits to our message, we will have to revise the value of our parity bit again to make sure that the parity is even.

As a convention (a choice mathematicians make together, for no other reason than consistency), all valid messages must have even parity, and this is accomplished by the attachment of the parity bit.

Consider the group of 8 digits below:  
Does the entire grid of 8 digits have even parity?

Was there an error somewhere in this grid?

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 |

Consider the groups of 8 digits below:

Does the group have even parity? Consider all 8 digits together.

Was there an error somewhere in these two rows?

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 0 | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 0 | 0 |

Go to the Math Hub and observe the applet titled “Original Geogebra Applet 3”.

<http://5010.mathed.usu.edu/Fall2020/BOgborn/Math%205010%20Assignments.html>

We’ll become very familiar with this applet and its contents, but right now it will seem a tad overwhelming. But don’t worry, we don’t need to understand everything that is going on in the applet right now. We are here only to practice parity.

Take a look at cells B5, C5, and D5. The contents of these cells are binary strings, they are codes. B5 and D5 have a leading 5 so that the software will display any leading zeros that the code may start with, so ignore the 5s.

Write the binary code (no 5s) found in cells B5, C5, and D5 from left to right in the space below:

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_

Notice that the code above has a red parity bit. That means that the parity of the entire string should be even if an error has not occurred.

Now we can answer question number 2 in the applet. Write your answer to question number 2 below:

2:

How many 1s:

Even or Odd:

Parity:

Valid Message?:

Now that we’ve answered the question about this code, I want to turn your attention to the colored grid in cells F2 to I5 (a square grid). This grid has 16 cells, one cell for each digit in the code we worked with for problem 2 just a moment ago. We’ll learn more about this grid next time, but right now we are focused on practicing parity.

Answer questions 3 and 4 on the applet that deal with parity of certain parts of the grid. Provide your answers below and on the next page if necessary:

3: 4:

How many 1s: How many 1s:

Even or Odd: Even or Odd:

Parity: Parity:

Valid Message?: Valid Message?:

If you counted correctly on questions 2,3, and 4, you should notice that each time, there was an even number of 1s, which indicates even parity, which indicates no error and a valid message. This is not a coincidence, because the applet that you have been looking at automatically generates the parity bits for these binary strings.

This applet is set up to take in user input of 11 binary digits, just the right amount for your chosen time of day. Try following the first two instructions to input your time of day. Does the red parity bit number match the parity bit you calculated earlier for your time of day code?