2. Show that the first 100 primes greater 3 are either of the form 6k + 1 or 6k + 5.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.2</title>

<script type="text/javascript">

function primeProof(){

var primes = [];

var ul = document.createElement('ul');

var count = 0;

while(count < 100){

primes = addPrime(primes);

if(primes[primes.length - 1] > 3){

count++;

}

}

var less3 = true;

var pos = 0;

while(less3){

if (primes[pos] < 3) {

pos++;

}

else if(primes[pos] == 3){

pos++;

less3 = false;

}

else{

console.log("3 not in array");

}

}

var proof = true;

var toPrint = "";

for(i = 0; i < 100; i++){

if(!plus1Form(primes[pos])){

if(!plus5Form(primes[pos]) && plus5Form(primes[pos]) != 0){

toPrint = toPrint + "<br/>Proof is false";

proof = false;

}

else{

toPrint = toPrint + "<br/>6(" + plus5Form(primes[pos]) + ")+5 = " + primes[pos];

}

}

else{

toPrint = toPrint + "<br/>6(" + plus1Form(primes[pos]) + ")+1 = " + primes[pos];

}

pos++;

}

document.write("The proof is " + proof);

document.write(toPrint);

}

function addPrime(array){

if(array.length == 0){

return [1];

}

var num = array[array.length - 1] + 1;//Next Num

var find = true;

if(num == 2){

find = false;

array.push(num);

}

else if(num%2 == 0){

num++;

}

while(find){

var div = num - 1;

var flag = true;

while(flag && div > 1){

if(num%div == 0){

flag = false;

}

else{

div--;

}

}

if(div == 1){

find = false;

array.push(num);

}

else{

num = num + 2;//speed up the process

}

}

return array;

}

function plus1Form(num){

var test = 0;

while(test < num){

if(((6\*test) + 1) == num){

return test;

}

else{

test++;

}

}

return false;

}

function plus5Form(num){

var test = 0;

while(test < num){

if(((6\*test) + 5) == num){

return test;

}

else{

test++;

}

}

return false;

}

</script>

</head>

<body onload="primeProof()">

<h1>Prime Number Proof</h1><br/>

</body>

</html>

4. Support the claim that 12 + 32 + 52 + … + (2n + 1)2 =  , by showing that it is true for all n ≤ 100. Now try to prove it.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.4</title>

<script type="text/javascript">

function oddSquares(){

var proof = true;

for(var i = 1; i <= 100; i++){

if(longWay(i) != shortWay(i)){

proof = false;

var failed = i;

}

}

if(proof){

document.write("The proof is true for numbers 1 - 100");

}

else{

document.write("The proof is false by the number " + failed);

}

}

function longWay(num){

var sum = 0;

for(var i = num; i > 0; i--){

sum = sum + (((2\*i) - 1) \* ((2\*i) - 1));

}

return sum;

}

function shortWay(num){

return (num\*((4\*(num\*num))-1))/3;

}

</script>

</head>

<body onload="oddSquares()">

<h1>Odd Square Sums</h1><br/>

</body>

</html>

6. Use the chart method to find the sum of the first n fourth powers. Test it for all n ≤ 100.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.6</title>

<script type="text/javascript">

function chartMethod(){

document.write("<table><tr><td>Number&nbsp</td><td>4Power&nbsp</td><td>Current Sum</td></tr>");

var sum = 0;

for(var i = 1; i <= 100; i++){

document.write("<tr><td>" + i + "</td>");

var fourth = i\*i\*i\*i;

document.write("<td>" + fourth + "</td>");

sum = sum + fourth;

document.write("<td>" + sum + "</td></tr>");

}

}

</script>

</head>

<body onload="chartMethod()">

<h1>The Chart Method</h1><br/>

</body>

</html>

8. Repeat the previous exercise using the summation laws given by formulas (1) and (2). How will you handle cn + d? Which program is faster for large values of N?

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.8</title>

<script type="text/javascript">

function formula(){

//Variables are ment to be changed by the user

var sum = 0;

var a = 4;

var b = 3;

var c = 2;

var d = 1;

var n = 100;

for(var i = 1; i <= n; i++){

var total = a\*power3(i) + b\*power2(i) + c\*i + d;

sum = sum + total;

}

document.write("The sum is " + sum);

}

function power3(n){

return ((n\*(n+1))/2) \* ((n\*(n+1))/2);

}

function power2(n){

return((n+1)\*((2\*n)+1)\*n)/2

}

</script>

</head>

<body onload="formula()">

<h1>Formula</h1><br/>

</body>

</html>

“cn” is handled by simply multiplying c\*n and therefore no need for a function to solve this problem. “d” gets handled as itself as well.

This program is faster than the one in problem 7 since the computer only needs to compute one equation instead of looping through the same operation on different numbers.

10. Show that none of the first 100 triangular numbers are the sum of two consecutive squares.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.8</title>

<script type="text/javascript">

function triSquares(){

squares = [];

squares = addSquare(squares);

var proof = true;

var n = 1;

var lastTri = 0;

while(proof && n<=100){

//Find the next triangle number

var tri = n + lastTri;

while(addSquare[addSquare.length-1] < tri){

squares = addSquare(squares);

}

var sum = false;

var index = 1;

while(!sum && squares[index] < tri){

if((squares[index] + squares[index - 1]) == tri){

sum = true;

proof = false;

document.write("The proof is false " + squares[index] + " + " + squares[index - 1]);

}

else{

index++;

}

}

n++;

}

if(proof){

document.write("The proof is true");

}

}

function addSquare(array){

var square;

var last = array.length + 1;//the next oblong number to be added (skip 1)

square = last\*last;

array.push(square);

return array;

}

function addTriangle(array){

if(array.length == 0){

array.push(1);

}

else{

var toAdd;

toAdd = array[array.length - 1] + (array.length + 1);

array.push(toAdd);

}

return array;

}

</script>

</head>

<body onload="triSquares()">

<h1>100 Triangles</h1><br/>

</body>

</html>

12. Show that the only cubes among the Fibonacci numbers in the previous exercise are 1 and 8.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.12</title>

<script type="text/javascript">

function cubedFibo(){

var cubes = [];

cubes = addCube(cubes);

var fiboCubes = [];

var fibos = [];

//Find Fibo numbers

for(var i = 0; i < 50; i++){

var num = findFibo(i,fibos);

fibos.push(num);

while(cubes[cubes.length - 1] < num){

cubes = addCube(cubes);

}

if(cubes[cubes.length - 1] == num){

fiboCubes.push(cubes[cubes.length - 1]);

}

}

document.write("The cubed Fibonacci numbers are " + fiboCubes.toString());

}

function addCube(array){

var cube;

var last = array.length + 1;//the next oblong number to be added (skip 1)

cube = last\*last\*last;

array.push(cube);

return array;

}

function findFibo(num, cached){

if(num == 0){

return 1;

}

else if(num == 1){

return 1;

}

else{

return cached[num - 1] + cached[num - 2];

}

}

</script>

</head>

<body onload="cubedFibo()">

<h1>Cubed Fibonacci</h1><br/>

</body>

</html>

14. Verify that for *k* ≤ 100, *u*3*k* is even while the rest are odd. Then prove this for all values of *k*.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.14</title>

<script type="text/javascript">

//ONLY WORKS UP TO 77TH fibo(CHANGE LIMIT TO 76)

function evenFibo(){

var fibos = [];

var proof = true;

var i = 0;

while(proof && i <= 77){

var num = findFibo(i, fibos);

fibos.push(num);

if((i + 1)%3 == 0){

if(num%2 == 1){

proof = false;

console.log("Not all are even");

}

}

else{

if(num%2 == 0){

proof = false;

console.log("The rest are not odd");

}

}

i++;

}

console.log(fibos.toString());

console.log("The proof is " + proof);

}

function findFibo(num, cached){

if(num == 0){

return 1;

}

else if(num == 1){

return 1;

}

else{

return cached[num - 1] + cached[num - 2];

}

}

</script>

</head>

<body onload="evenFibo()">

<h1>Even Fibonacci</h1><br/>

</body>

</html>

16. Verify that for *k* ≤ 100, we have *u*2 + *u*4 + *u*6 + … + *u*2k = *u*2k+1 - 1.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Question 2.16</title>

<script type="text/javascript">

//ONLY WORKS UP TO 77TH fibo(CHANGE LIMIT TO 76)

function twoFibos(){

var currentSum = 0;

var fibos = [];

var proof = true;

var i = 0;

while(proof && i <= 77){

var num = findFibo(i, fibos);

fibos.push(num);

if((i + 1)%2 == 0){

currentSum = currentSum + num;

}

else{

if(i > 2){

if(currentSum != num - 1){

proof = false;

document.write("The sum is not equal");

}

}

}

i++;

}

document.write("The proof is " + proof);

}

function findFibo(num, cached){

if(num == 0){

return 1;

}

else if(num == 1){

return 1;

}

else{

return cached[num - 1] + cached[num - 2];

}

}

</script>

</head>

<body onload="twoFibos()">

<h1>Even Fibonacci</h1><br/>

</body>

</html>

18. \* Verify that for *n* ≤ 100, we have *u*n2 - *u*n+1*u*n-1 = (-1)n-1.