MAGMA Genus Calculator Program

```
CalculateGenus:=procedure (invariants)
/*
    Uses Maclachlan's formula to calculate the genus of a group based on its invariants.
*/

    order:=1;       //Order
    s:=#invariants;  //Rank

    //Calculate group order
    for i in [1..s] do
        order := order*invariants[i];
    end for;

    //Check that invariants are in proper canonical form
    canonical := true;
    i := 1;
    while (i lt s) do
        remainder := invariants[i+1] mod invariants[i];
        if (remainder eq 0) then i := i + 1;
        else i := s;
            canonical := false;
            print "\tOrder:", order, "   Genus: UNKNOWN   Invariants: NOT CANONICAL FORM";
        end if;
    end while;

     //If the group is in canonical form, calculate genus
     if (canonical) then
        if (s eq 1) then
            print "Order:", order, "   Genus:", 0, "   Invariants:", invariants;
        elif (s eq 2) then
            print "\tOrder:", order, "   Genus:", 1, "   Invariants:", invariants;
        else
            gamma:=Truncate(s/2);
            minimize := [];  //Right hand side array

            //Calculate the right hand side results
            for y in [0..gamma] do
                j:=s-2*y;
                if (j eq 0) then
                    sum := 0;
                else
                    sum := 1 - 1/invariants[j];

                     //Calculate the sumation within the formula
                    for i in [1..j] do
                        sum := sum + 1 - (1/invariants[i]);
                    end for;
                end if;
                minimize[y+1] := 2*(y-1) + sum;
            end for;

            //Find minimum right hand side to use in formula
            minimum := Min(minimize);
```

```
            //Find the genus for the group
            genus := minimum * order /2 + 1;

            //Find the gamma value for the group
            for y in [0..gamma] do
                if (minimize[y+1] eq minimum) then gammavalue := y;
                end if;
            end for;

            print "\tOrder:", order, "   Genus:", genus, "   Invariants:", invariants;
        end if;
    end if;
end procedure;



FindAbelianGroups := procedure (order)
/*
    Prints the Magama Small Group number of every abelian group of the given order.
*/

    numGroups := NumberOfSmallGroups(order);

    for group in [1..numGroups] do
        G := SmallGroup (order,group);

        //If the group is abelian, print group number
        if IsAbelian (G) then
            print "Magma Small Group (", order, ",", group, ")";
        end if;
    end for;
end procedure;



AbelianGroupsOfOrder256p := procedure (order)
/*
Since there are so many groups order 256*p in the Small Group Library in Magma, this program hangs up
when asked to run through every one. Therefore, the abelian groups of order 256, 768, 1280, and 1792 were
found ahead of time using the FindAbelianGroups procedure, and now each group is individually sent to the
CalculateGenus procedure.
*/

//Currently groups with genus 0 and 1 are set to not print!!

    //G := SmallGroup(order, 1);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 1)";
    //CalculateGenus(I);

    //G := SmallGroup(order, 39);
    //I := Invariants(G);
```

MAGMA Genus Calculator Program

```
//print "Magma Small Group (", order, ", 39)";
//CalculateGenus(I);

//G := SmallGroup(order, 316);
//I := Invariants(G);
//print "Magma Small Group (", order, ", 316)";
//CalculateGenus(I);

//G := SmallGroup(order, 497);
//I := Invariants(G);
//print "Magma Small Group (", order, ", 497)";
//CalculateGenus(I);

//G := SmallGroup(order, 537);
//I := Invariants(G);
//print "Magma Small Group (", order, ", 537)";
//CalculateGenus(I);

G := SmallGroup(order, 826);
I := Invariants(G);
print "Magma Small Group (", order, ", 826)";
CalculateGenus(I);

G := SmallGroup(order, 4384);
I := Invariants(G);
print "Magma Small Group (", order, ", 4384)";
CalculateGenus(I);

G := SmallGroup(order, 5525);
I := Invariants(G);
print "Magma Small Group (", order, ", 5525)";
CalculateGenus(I);

G := SmallGroup(order, 6534);
I := Invariants(G);
print "Magma Small Group (", order, ", 6534)";
CalculateGenus(I);

G := SmallGroup(order, 6723);
I := Invariants(G);
print "Magma Small Group (", order, ", 6723)";
CalculateGenus(I);

G := SmallGroup(order, 6732);
I := Invariants(G);
print "Magma Small Group (", order, ", 6732)";
CalculateGenus(I);

G := SmallGroup(order, 10298);
I := Invariants(G);
print "Magma Small Group (", order, ", 10298)";
CalculateGenus(I);
```

```
    G := SmallGroup(order, 13313);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 13313)";
    CalculateGenus(I);

    G := SmallGroup(order, 26308);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 26308)";
    CalculateGenus(I);

    G := SmallGroup(order, 26959);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 26959)";
    CalculateGenus(I);

    G := SmallGroup(order, 26973);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 26973)";
    CalculateGenus(I);

    G := SmallGroup(order, 53038);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 53038)";
    CalculateGenus(I);

    G := SmallGroup(order, 55608);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 55608)";
    CalculateGenus(I);

    G := SmallGroup(order, 55626);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 55626)";
    CalculateGenus(I);

    G := SmallGroup(order, 56059);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 56059)";
    CalculateGenus(I);

    G := SmallGroup(order, 56082);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 56082)";
    CalculateGenus(I);

    G := SmallGroup(order, 56092);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 56092)";
    CalculateGenus(I);

end procedure;
```

MAGMA Genus Calculator Program

```
AbelianGroupsOfOrder512p := procedure(order)
/*
    Since there are so many groups order 512*p in the Small Group Library in Magma, this program hangs up
    when asked to run through every one. Therefore, the abelian groups of order 512 and 1536 were found ahead
    of time using the FindAbelianGroups procedure, and now each group is individually sent to the CalculateGenus
    procedure.
*/

//Currently groups with genus 0 and 1 are set to not print!!

    //G := SmallGroup(order, 1);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 1)";
    //CalculateGenus(I);

    //G := SmallGroup(order, 859);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 859)";
    //CalculateGenus(I);

    //G := SmallGroup(order, 1699);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 1699)";
    //CalculateGenus(I);

    //G := SmallGroup(order, 2009);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 2009)";
    //CalculateGenus(I);

    //G := SmallGroup(order, 2040);
    //I := Invariants(G);
    //print "Magma Small Group (", order, ", 2040)";
    //CalculateGenus(I);

    G := SmallGroup(order, 2046);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 2046)";
    CalculateGenus(I);

    G := SmallGroup(order, 29399);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 29399)";
    CalculateGenus(I);

    G := SmallGroup(order, 33712);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 33712)";
    CalculateGenus(I);

    G := SmallGroup(order, 56686);
    I := Invariants(G);
    print "Magma Small Group (", order, ", 56686)";
```

```
CalculateGenus(I);

G := SmallGroup(order, 58942);
I := Invariants(G);
print "Magma Small Group (", order, ", 58942)";
CalculateGenus(I);

G := SmallGroup(order, 60616);
I := Invariants(G);
print "Magma Small Group (", order, ", 60616)";
CalculateGenus(I);

G := SmallGroup(order, 60895);
I := Invariants(G);
print "Magma Small Group (", order, ", 60895)";
CalculateGenus(I);

G := SmallGroup(order, 87977);
I := Invariants(G);
print "Magma Small Group (", order, ", 87977)";
CalculateGenus(I);

G := SmallGroup(order, 260300);
I := Invariants(G);
print "Magma Small Group (", order, ", 260300)";
CalculateGenus(I);

G := SmallGroup(order, 387089);
I := Invariants(G);
print "Magma Small Group (", order, ", 387089)";
CalculateGenus(I);

G := SmallGroup(order, 400018);
I := Invariants(G);
print "Magma Small Group (", order, ", 400018)";
CalculateGenus(I);

G := SmallGroup(order, 419735);
I := Invariants(G);
print "Magma Small Group (", order, ", 419735)";
CalculateGenus(I);

G := SmallGroup(order, 420501);
I := Invariants(G);
print "Magma Small Group (", order, ", 420501)";
CalculateGenus(I);

G := SmallGroup(order, 420515);
I := Invariants(G);
print "Magma Small Group (", order, ", 420515)";
CalculateGenus(I);

G := SmallGroup(order, 6249624);
```

MAGMA Genus Calculator Program

```
I := Invariants(G);
print "Magma Small Group (", order, ", 6249624)";
CalculateGenus(I);

G := SmallGroup(order, 6276915);
I := Invariants(G);
print "Magma Small Group (", order, ", 6276915)";
CalculateGenus(I);

G := SmallGroup(order, 7529607);
I := Invariants(G);
print "Magma Small Group (", order, ", 7529607)";
CalculateGenus(I);

G := SmallGroup(order, 7532375);
I := Invariants(G);
print "Magma Small Group (", order, ", 7532375)";
CalculateGenus(I);

G := SmallGroup(order, 7532393);
I := Invariants(G);
print "Magma Small Group (", order, ", 7532393)";
CalculateGenus(I);

G := SmallGroup(order, 10481222);
I := Invariants(G);
print "Magma Small Group (", order, ", 10481222)";
CalculateGenus(I);

G := SmallGroup(order, 10493039);
I := Invariants(G);
print "Magma Small Group (", order, ", 10493039)";
CalculateGenus(I);

G := SmallGroup(order, 10493062);
I := Invariants(G);
print "Magma Small Group (", order, ", 10493062)";
CalculateGenus(I);

G := SmallGroup(order, 10494174);
I := Invariants(G);
print "Magma Small Group (", order, ", 10494174)";
CalculateGenus(I);

G := SmallGroup(order, 10494201);
I := Invariants(G);
print "Magma Small Group (", order, ", 10494201)";
CalculateGenus(I);

G := SmallGroup(order, 10494213);
I := Invariants(G);
print "Magma Small Group (", order, ", 10494213)";
CalculateGenus(I);
```

```
end procedure;


AbelianGroupsOfOrder1024 := procedure(order)
/*
Since the Small Group Library in Magma does not include groups of size 1024, the abelian groups of that size
were found manually, then converted into Canonical Form to obtain the invariants. Then each group is
individually sent to the CalculateGenus procedure. These groups have no Magma Small Group number.
*/

//Currently groups of order 1 and 0 are set to not print!

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([1024]);

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([2,512]);

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([4,256]);

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([8,128]);

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([16,64]);

    //print "Magma 1024 Group (",order, ")";
    //CalculateGenus([32,32]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,256]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,4,128]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,8,64]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,16,32]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([4,4,64]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([4,8,32]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([4,16,16]);
```

MAGMA Genus Calculator Program

```
print "Magma 1024 Group (",order, ")";
CalculateGenus([8,8,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,2,128]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,4,64]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,8,32]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,16,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,4,4,32]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,4,8,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,8,8,8]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([4,4,4,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([4,4,8,8]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,2,2,64]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,2,4,32]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,2,8,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,4,4,16]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,4,8,8]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,4,4,4,8]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([4,4,4,4,4]);

print "Magma 1024 Group (",order, ")";
CalculateGenus([2,2,2,2,2,32]);
```

```
    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,4,16]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,8,8]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,4,4,8]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,4,4,4,4]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,2,16]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,4,8]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,4,4,4]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,2,2,8]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,2,4,4]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,2,2,2,4]);

    print "Magma 1024 Group (",order, ")";
    CalculateGenus([2,2,2,2,2,2,2,2,2,2]);

end procedure;



GroupGenusCalculator:=procedure(minOrder, maxOrder)
/*
 Finds every abelian group within the range of orders provided. Then calculates and prints the Magma Small
Group Number, order, genus, and invariants for each of those abelian groups.
*/

    //Currently only groups with genus greater than 1 are printed!

    for order in [minOrder..maxOrder] do

        //Checks if the order needs to be handled manually
        case order:
            when 256: AbelianGroupsOfOrder256p(256);
            when 512: AbelianGroupsOfOrder512p(512);
            when 768: AbelianGroupsOfOrder256p(768);
```

```
                when 1024: AbelianGroupsOfOrder1024(1024);
                when 1280: AbelianGroupsOfOrder256p(1280);
                when 1792: AbelianGroupsOfOrder256p(1792);
                when 1536: AbelianGroupsOfOrder512p(1536);
                else
                    //Use Small Group Library to find all groups of this order
                    numGroups := NumberOfSmallGroups(order);

                    for group in [1..numGroups] do
                        G := SmallGroup (order,group);

                        //If the group is abelian, calculate the invariants and genus
                        if IsAbelian (G) then
                            I := Invariants(G);

//Remove the following if statement to print groups of order 0 and 1!!
                            //Checks if group is of rank 3 or more.
                            if (#I gt 2) then
                                print "Magma Small Group (", order, ",", group, ")";
                                CalculateGenus(I);
                            end if;
                        end if;
                    end for;
            end case;
        end for;
end procedure;
```