Proceedings of Science and Mathematics
Faculty of Science, Universiti Teknologi Malaysia
Vol. 24, 2024, page 154-164

The Computation of Some Properties of Finite Abelian Groups by Using C++ Programming

Nur Alya Izzati Baharim, Nor Muhainiah Mohd Ali
Department of Mathematics, Faculty of Science, Universiti Teknologi Malaysia,
81310 Skudai, Johor Darul Takzim
Corresponding author: normuhainiah@utm.my

Abstract
An abelian group is a group in which the operation is commutative. Examples of finite abelian groups include the group of integers under addition modulo *n* and the group of integers under multiplication modulo *n,* where *n* is any positive integer. These groups play a crucial role in various areas of mathematics and computer science due to their structural properties and applications. As *n* increases, calculating properties such as the order of group, the order and inverse of each element, cyclic subgroups, generators, and lattice diagrams becomes increasingly complex and time-consuming. Consequently, a specialized program is necessary for these computations to ensure accuracy and efficiency. In this research, a program is developed using Microsoft Visual C++ that allows users to input any positive integer up to 1000 to generate answers for these group properties. This tool significantly simplifies the process of analyzing complex group structures, making it accessible for researchers and educators alike.
Keywords Finite Abelian Groups; Group of Integers Under Addition Modulo *n*; Group of Integers Under Multiplication Modulo *n*; Microsoft Visual C++

1. Introduction

Algebra is one of the fundamental subjects in Mathematical sciences and group theory is one of the topics which have been studied since the 19th century [1]. Group characteristics have been the subject of a lot of discussion over the years. This includes the elements and their inverses, the cyclic and non-cyclic, the nature of groups, and the study of their lattice diagrams. These have covered a wide range of group types, including finite groups and some special cases such as dihedral groups. The additive group of integers modulo *n* is represented by the symbol $\mathbb{Z}_n$. In mathematical terms, $\mathbb{Z}_n$ is the set of equivalence classes of integers under the relation of congruence modulo *n*. If the difference between two integers is divisible by *n*, these numbers are said to be congruent modulo *n*. The symbol *U*(*n*) represents the group of units modulo *n*, also known as the multiplicative group of integers modulo *n*. The integers in the set $\mathbb{Z}_n$ that are relatively prime to *n*, or that have no shared factors other than 1 with *n*, are the elements of *U*(*n*).

Manually, to generate some properties of $\mathbb{Z}_n$ and *U*(*n*) which include the elements of group, order and the unique inverse of each element and also the lattice diagram will cost a lot of time when the value of *n* increases. Therefore, a new method by using software is needed to reduce this difficulty. Previously, Mohd Ali et al. [2] have developed a C++ program interface to display the properties of $\mathbb{Z}_n$ and *U*(*n*). However, the input of *n* for the program is limited to positive values of $n \leq 120$ and all the properties are shown in one interface. Later in 2015, Mohd Ali et al. [3] upgraded this program limitation until $n \leq 200$ and improvised the interface layout. The programme limitations of groups $\mathbb{Z}_n$ and *U*(*n*) have been modified by Abd Rahman [4] in 2019 to $n \leq 400$ and the dihedral group, D_n, where $n = 3, 4,$ and 5. Then, this report is to upgrade the program by improvised the limitation of group $\mathbb{Z}_n$ and *U*(*n*) up to $n = 1000$.

2. Literature Review

2.1 Programming Involving Group Theory

Mohd Ali et al. [2] have developed a program which can display some properties of some finite abelian groups. The program allowed the user to choose the integer n for $n \leq 120$. The order of group, order and inverse of each element, cyclic subgroups, and list of all generators of a group can be computed by using this program. In this program, the authors also developed the coding so that the program can display the lattice diagram of the cyclic groups. All the properties are displayed in one interface after a group has been chosen.

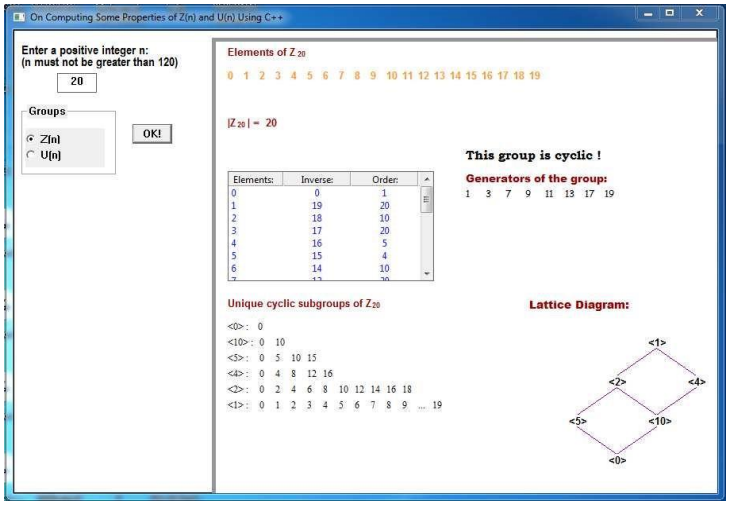

Figure 1 Interface for $\mathbb{Z}_{20}$

Abd Rahman [4] have developed a program to determine all elements of the group, order of the group, inverse and order of each element, generators of the group, cyclic subgroups as well as the lattice diagrams of group $\mathbb{Z}_n$ and cyclic group $U(n)$ for $n \leq 400$. This program also included some properties for some dihedral groups, D_n which can display the symmetry shape of the n-gon of the groups and their Cayley table. The properties can be reviewed by selecting one of the buttons and entering the desired value for n.

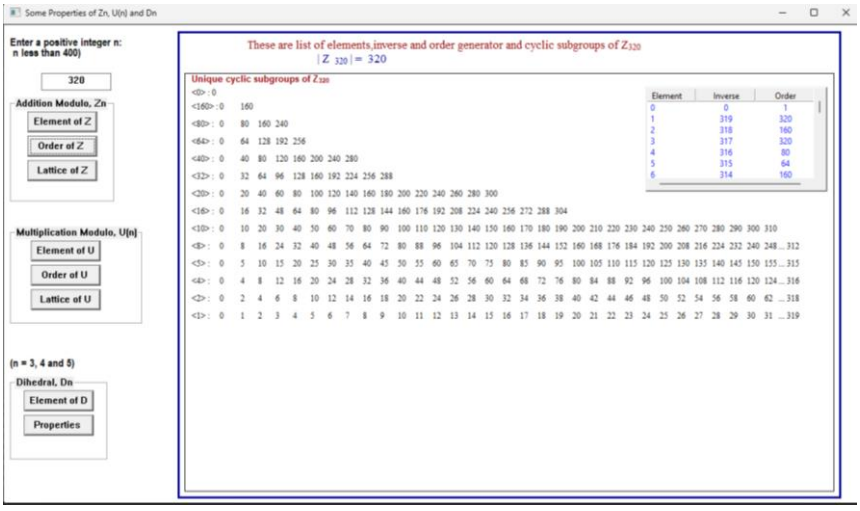

Figure 2 Interface for $\mathbb{Z}_{320}$

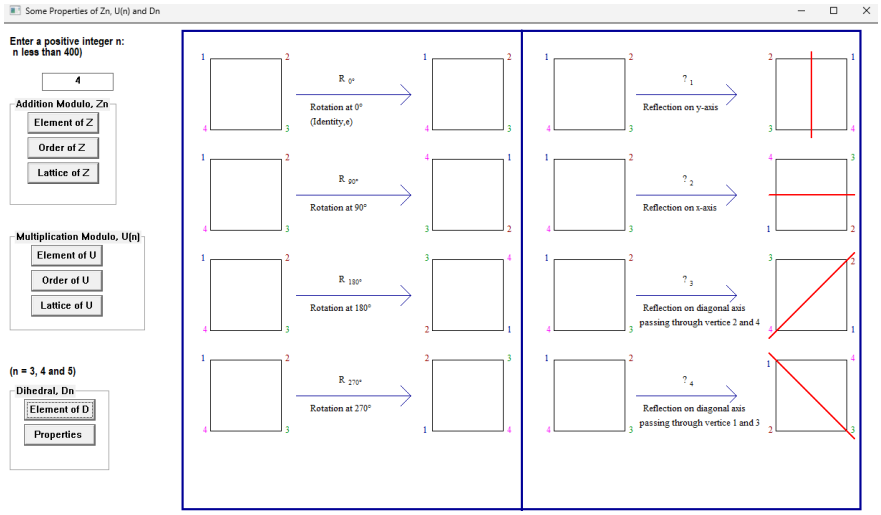

Figure 3 Interface for D_4

2.2 Some Basic Concepts in Group Theory

In this section, the definition of a group that will be covered in this research including the order of group and their elements, cyclic group, cyclic subgroup, and lattice diagram and the definition for the groups $\mathbb{Z}_n$ and $U(n)$ are discussed.

Definition 2.2.1 [8] The number of elements of a group (finite or infinite) is called the **order of a group**. The notation $|G|$ is used to denote the order of G.

Definition 2.2.2 [8] The **order of an element** g in a group G is the smallest positive integer n such that $g^n = e$ (In additive notation, this would be $ng = 0$). The order of an element g is denoted by $|g|$.

Definition 2.2.3 [8] Let $a \in G$. Then $\langle a \rangle = \{ a^n \mid n \in \mathbb{Z} \} = \{e, a, a^2, a^3, ...\}$ is called a **cyclic subgroup** of G generated by a.

Definition 2.2.4 [8] Let $a \in G$. Then $\langle a \rangle = \{ a^n \mid n \in \mathbb{Z} \} = \{..., a^{-2}, a^{-1}, e, a, a^2, a^3, ...\}$. If $G = \langle a \rangle$, then G is called as a **cyclic group** generated by a.

Definition 2.2.5 [8] Let $\{H_1, H_2, ..., H_n\}$ be a collection of subgroups of G and denoted by E the trivial subgroups of G with only one element. Then one arranges the groups vertically such that groups with higher order are placed nearer the top and groups with smaller order placed nearer the bottom

Definition 2.2.6 [8] The set $\mathbb{Z}_n = \{0, 1, ..., n\text{-}1\}$ for $n \geq 1$ is a group under addition modulo n. For any i in $\mathbb{Z}_n$ the inverse of i is $n - i$. This group is usually referred to as **the group of integers addition modulo *n***.

Example 2.2.1 [2] The elements of $\mathbb{Z}_6$ are 0, 1, 2, 3, 4 and 5. Hence the order of the group is 6. The computations of the order of the elements are as follows:

$|0| = 1$ since the order of the identity element is always 1.

$|1| = |5| = 6$ since $6 \times 1 = 6 \equiv 0$ and $6 \times 5 = 30 \equiv 0$.

$|2| = |4| = 4$ and $|3| = 2$.

One way of getting the inverse of each element is to use the formula $n - i$, where i is the element of 6. Therefore, $0^{-1} = 0$ (the inverse of the identity element is identity), $1^{-1} = 5$, $2^{-1} = 4$ and $3^{-1} = 3$. The elements 1 and 5 are the generators of this group since the order of those elements is the same as the order of $\mathbb{Z}_6$. The cyclic subgroups of $\mathbb{Z}_6$ are obtained by generating each element of the group. The following shows the cyclic subgroups of $\mathbb{Z}_6$:

$$\langle 0\rangle = \{0\},\ \langle 1\rangle = \langle 5\rangle = \mathbb{Z}_6,\ \langle 2\rangle = \langle 4\rangle = \{0, 2, 4\} \text{ and } \langle 3\rangle = \{0, 3\}$$

Hence the lattice diagram of $\mathbb{Z}_6$ is:

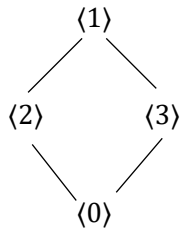

Figure 4: Lattice diagram of $\mathbb{Z}_6$

Definition 2.2.7 [8] For each $n > 1$, $U(n)$ is defined to be the set of all positive integers less than n and relatively prime to n. $U(n)$ is known as a **group of integers under multiplication modulo *n*.**

Example 2.2.2 [2] The elements of $U(6)$ consist of 1 and 5 only. Hence the order of the group is 2. The computations of the order of the elements are as follows:

$|1| = 1$ since the order of the identity element is always 1.

$|5| = 2$ since $5 \times 5 = 1$

The inverse of each element is: $1^{-1} = 1$ (the inverse of the identity element is identity, $5^{-1} = 5$. Generator of this group is 5. The cyclic subgroups of $U(6)$ are also obtained by generating each element of the group. The following shows the cyclic subgroups:

$$\langle 1\rangle = \{1\},\ \langle 5\rangle = \mathrm{U}(6)$$

Hence the lattice diagram of $U(6)$ is as follows of $U(6)$:

Figure 5: Lattice diagram of U(6)

3. Programming codes and the output related to the additive group of integers modulo *n*

3.1 Some programming codes. The codes below are used to minimize the potential for errors, C++ has adopted the convention of using header files to contain declarations. You make the declarations in a header file, then use the #include directive in every .cpp file or other header file that requires that declaration. The #include directive inserts a copy of the header file directly into the .cpp file prior to compilation.

```
#include <afxwin.h>
```

The codes below are to set the ID numbers for each event created for the interface. The ID numbers need to be integers and different from each event for the codes are not overlapped. These are two ID numbers added to the codes where IDC_BTNOPT_ElmtnOrdZ is the name for the ID and 501 is the ID number.

```
#define IDC_BTNOPT_ElmtnOrdZ        501
#define IDC_BTNOPT_ElmtnOrdU        506
```

To display the properties of the Abelian Groups that less than 1000, this code needs to be change to 1000 for the interface to run.

```
#define M 1000
```

The following code is to create the box that allow the user to write the value of n.

```
CEdit           enterN;
enterN.Create(WS_CHILD | WS_VISIBLE | WS_BORDER | SS_CENTER, CRect(CPoint(40, 70), CSize(60, 30)), this, IDC_ENTER_VALUE);
DrawRegion = CRect(CPoint(xMIN, yMIN), CPoint(xMAX, yMAX));
```

CEdit is a command to declare an edit box and enterN is the name of the variable. DrawRegion is an assign variable for the output area. enterN.Create to create the box with desired edit box using WS_CHILD. CRect(CPoint(40, 70), CSize(60, 30)) to change the size of the edit box.

Next, the following codes are used to create the button for displaying the elements, order, and the lattice diagram of $\mathbb{Z}_n$:

```
CButton   btnGrpGZ, btnElmtZ, btnOrdZ, btnLatZ;
btnGrpGZ.Create(L"ℤn", WS_CHILD | WS_VISIBLE | BS_GROUPBOX | WS_GROUP, CRect(CPoint(200, 10), CSize(250, 130)), this, IDC_BTNGRP_GROUPZ);
btnElmtnOrdZ.Create(L"Element, Inverse and Order of ℤn", WS_CHILD | WS_VISIBLE | BS_DEFPUSHBUTTON | BS_MULTILINE, CRect(CPoint(210, 30), CSize(230, 40)), this, IDC_BTNOPT_ElmtnOrdZ);
btnGenZ.Create(L"Generator of ℤn", WS_CHILD | WS_VISIBLE | BS_DEFPUSHBUTTON, CRect(CPoint(210, 75), CSize(230, 25)), this, IDC_BTNOPT_GenZ);
btnLatZ.Create(L"Lattice of ℤn", WS_CHILD | WS_VISIBLE | BS_DEFPUSHBUTTON, CRect(CPoint(210, 105), CSize(230, 25)), this, IDC_BTNOPT_LatZ);
```

CButton is a command to declare a button. btnGrpGZ is the name of the variable that assign to a button. With the name of the variable, btnGrpGZ.Create used to create the button. The name on the button can be edit at L"ℤn" with the desired type of button using BS_GROUPBOX. These commands applied to all buttons created for the interface.

Then, the event needs to be declared for each created button.

```
afx_msg void OnElement();
afx_msg void OnOrder();
afx_msg void OnLattice();
```

```
ON_BN_CLICKED(IDC_BTNOPT_ElmtZ, OnElement)
ON_BN_CLICKED(IDC_BTNOPT_OrdZ, OnOrder)
ON_BN_CLICKED(IDC_BTNOPT_LatZ, OnLattice)
```

OnElement() is the name of the event. Then, ON_BN_CLICKED used to declared the event in the event list with the name of the ID and event.

3.2 The output of addition modulo *n*, $\mathbb{Z}n$

3.2.1 Program Interface. Figure 3 illustrates the interface for the written program.

3.2.2 Some output of the program for $\mathbb{Z}_n$. Figure 4 shows the outputs displayed in the interface when the user keys in *n*=665 and selects the button "Element, Inverse and Order of $\mathbb{Z}n$".

The outputs on generator of $\mathbb{Z}_{665}$ shown in Figure 5 when the user select the button "Generator of $\mathbb{Z}n$".

The lattice diagram of $\mathbb{Z}_{665}$ is shown in Figure 6 when the user selects the button "Lattice Diagram of $\mathbb{Z}n$".

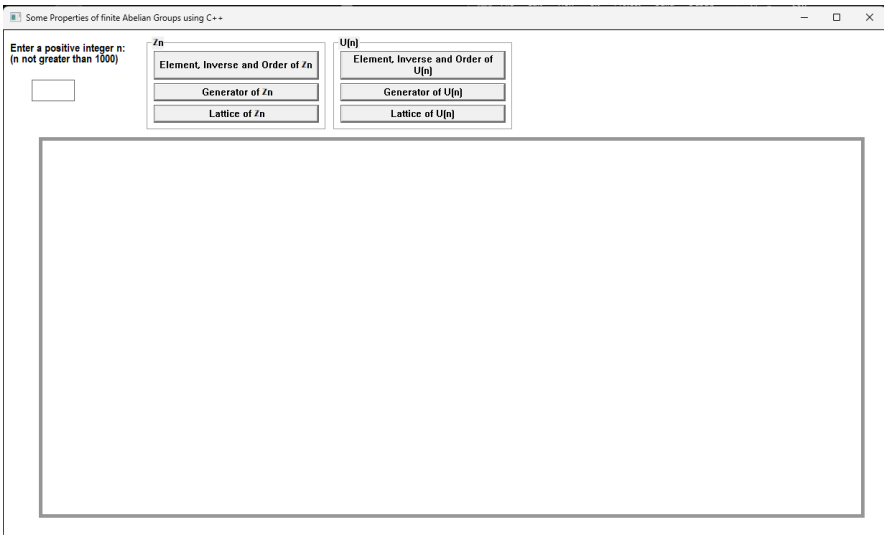

Figure 6

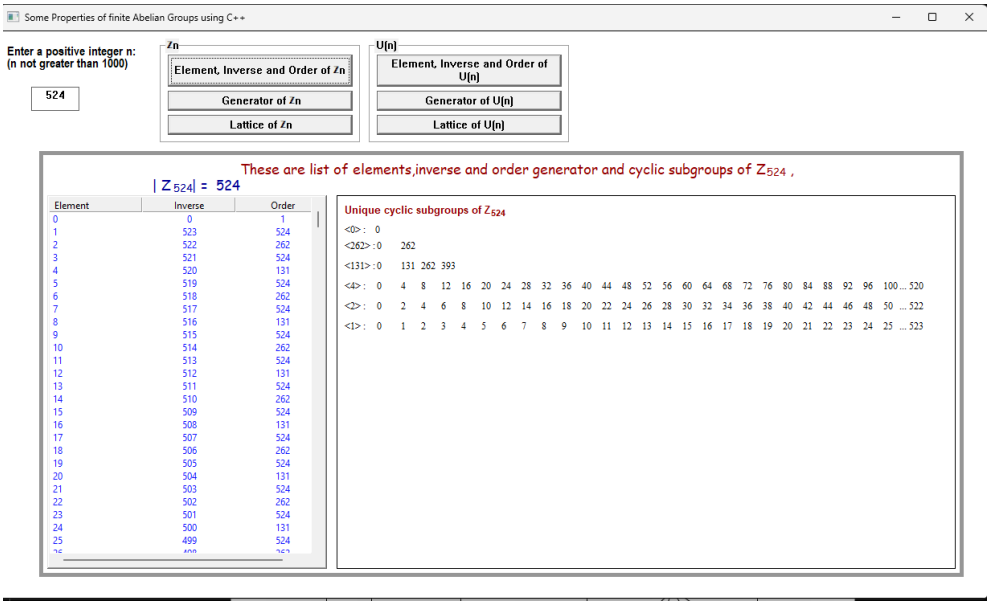

Figure 7

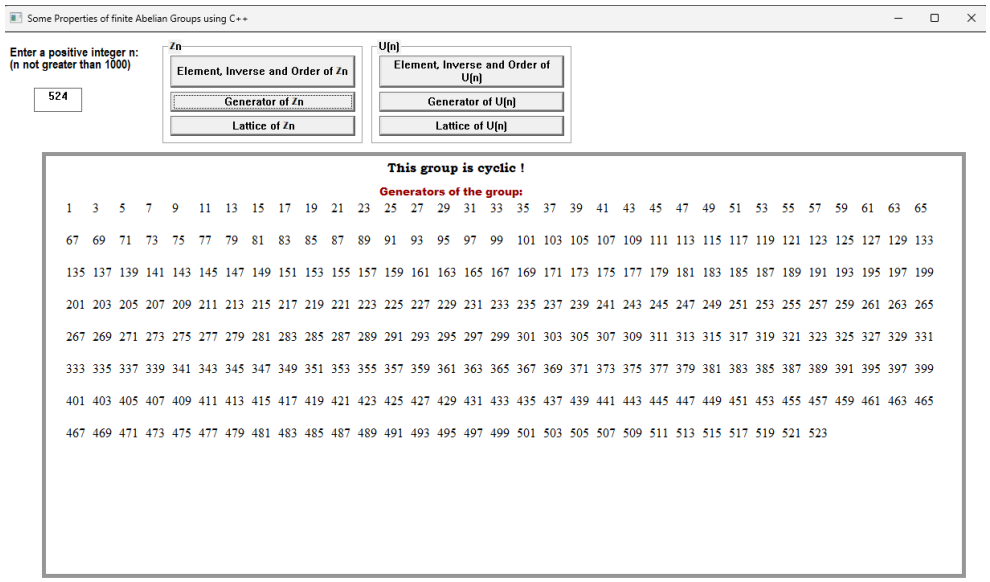

Figure 8

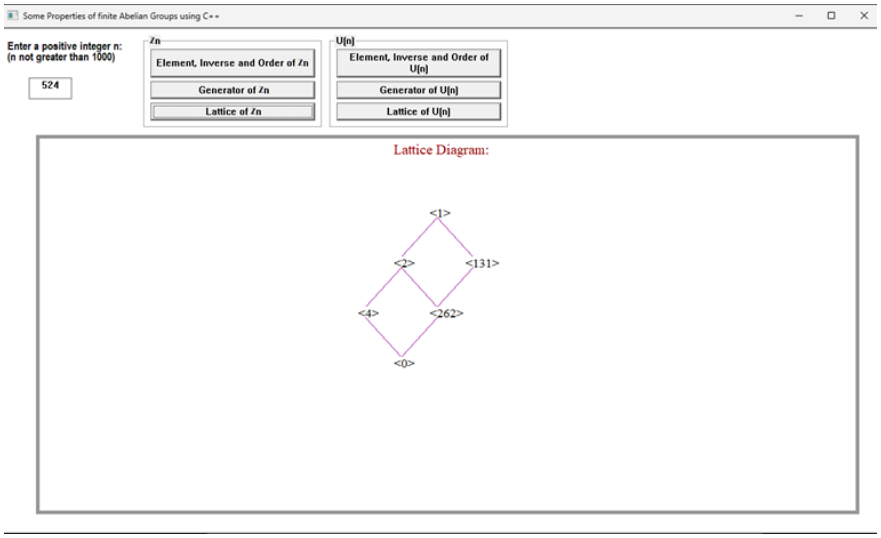

Figure 9

This program is written with message boxes that will appear in different situations. Since this program is limited for group of integers under addition modulo *n*, $\mathbb{Z}_n$ where *n* is any positive integer at most 1000, if the user accidently enters the value *n* less than 0 or greater than 1000 a message box warning as shown in Figure 7 will appear.

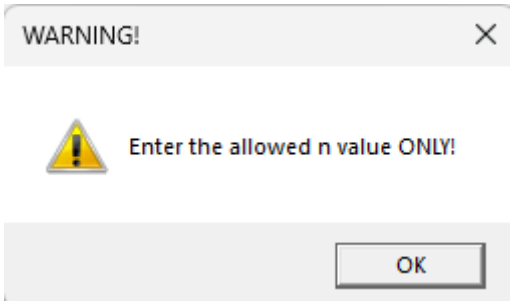

Figure 10

4. Programming codes and the output multiplicative group of integers modulo *n*

4.1 Some programming codes. The codes below are used to create the button for displaying the elements, order, and the lattice diagram of *U*(*n*):

```
btnGrpGU.Create(L"U(n)", WS_CHILD | WS_VISIBLE | BS_GROUPBOX | WS_GROUP,
CRect(CPoint(460, 10), CSize(250, 130)), this, IDC_BTNGRP_GROUPU);

btnElmtnOrdU.Create(L"Element, Inverse and Order of U(n)", WS_CHILD |
WS_VISIBLE | BS_DEFPUSHBUTTON | BS_MULTILINE, CRect(CPoint(470, 30),
CSize(230, 40)), this, IDC_BTNOPT_ElmtnOrdU);

btnGenU.Create(L"Generator of U(n)", WS_CHILD | WS_VISIBLE |
BS_DEFPUSHBUTTON, CRect(CPoint(470, 75), CSize(230, 25)), this,
IDC_BTNOPT_GenU);

btnLatU.Create(L"Lattice of U(n)", WS_CHILD | WS_VISIBLE | BS_DEFPUSHBUTTON,
CRect(CPoint(470, 105), CSize(230, 25)), this, IDC_BTNOPT_LatU);
```

4.2 The output of the program for U(n). Figure 8 shows the outputs displayed in the interface when the user keys in *n*=563 and selects the button “Element, Inverse and Order of U(n)”.

The outputs on generator of U(563) shown in Figure 9 when the user select the button “Generator of U(n)”.

The lattice diagram of U(563) is shown in Figure 10 when the user select the button “Lattice Diagram of U(n)”.

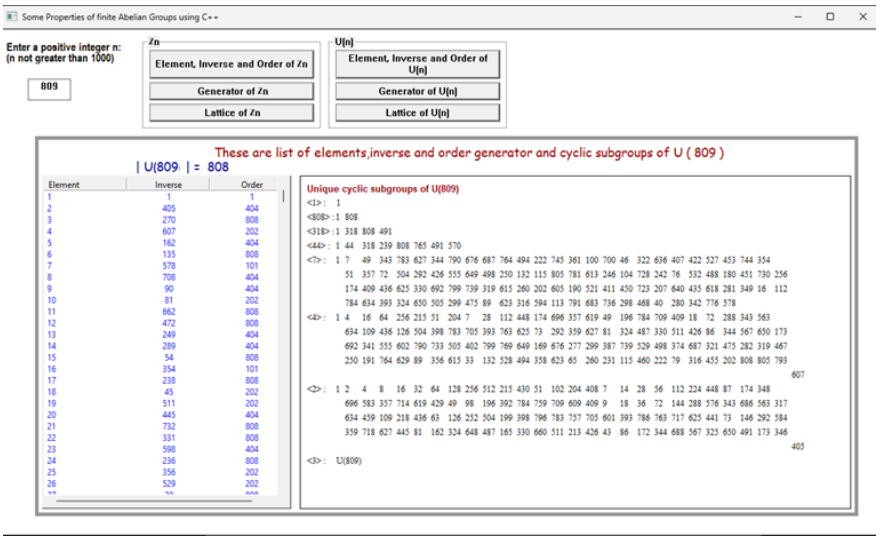

Figure 11

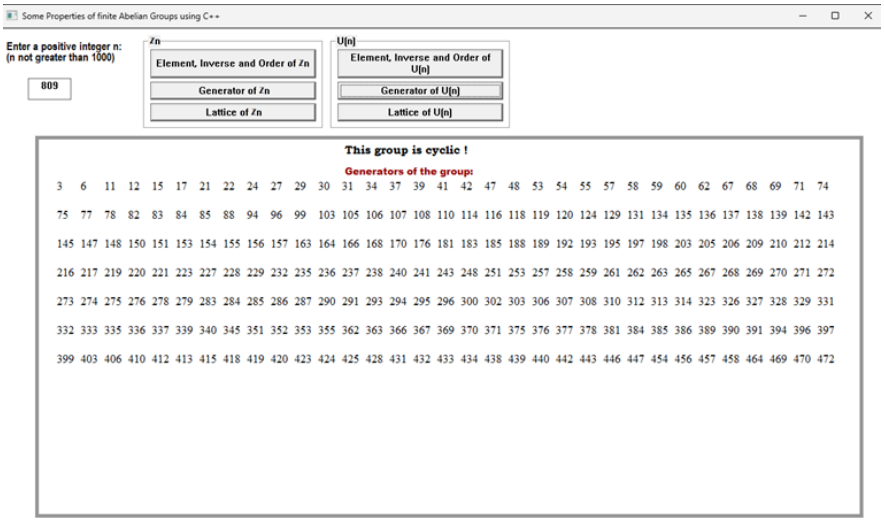

Figure 12

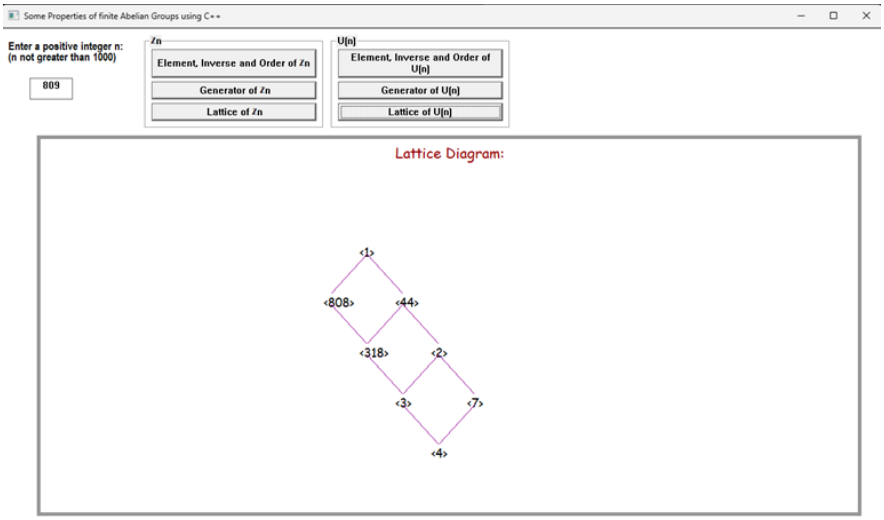

Figure 13

4.3 Message Boxes. This program is written with message boxes that will appear in different situations. Since this program is limited for group of integers under multiplication modulo n, $U(n)$ where n is any positive integer at most 1000, if the user accidently enters the value n less than 0 or greater than 1000 a message box warning as shown in Figure 11 will appear.

Another message box that is shown in Figure 12 informs the user about the display, that is, cyclic subgroups and lattice diagram will only be displayed if the chosen group is cyclic.

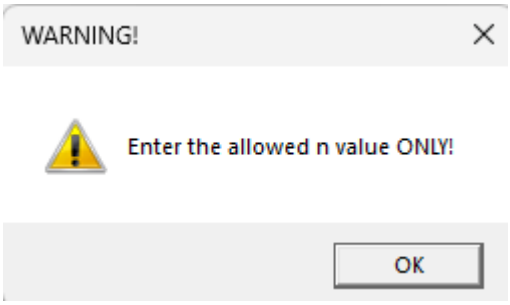

Figure 14

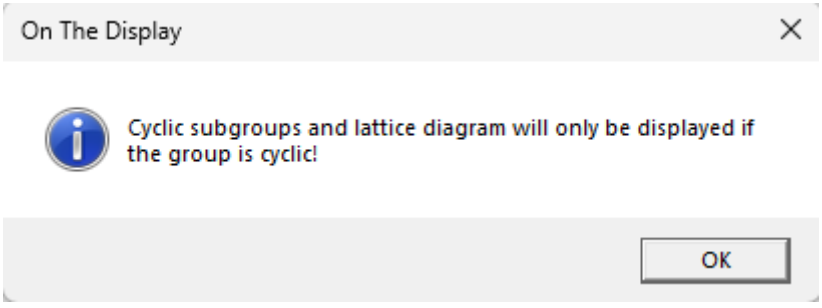

Figure 15

5. Conclusion

From this research, a program has been constructed to determine all elements of the group, order of the group, inverse and order of each element, generators of the group, cyclic subgroups as well

as the lattice diagrams of groups $\mathbb{Z}_n$ and cyclic group $U(n)$ for $n \leq 1000$. By entering the desired value of n and checking at one of the buttons, the properties will appear.

This program is hoped to be able to serve as a starting point for developing better and sophisticated programs.

6. Suggestion

The input of n for this program is limited to positive integers of n at most 1000. Therefore, my suggestions would be to eliminate this restriction on the value of n. Apart from that, an improved program to find the cyclic subgroups together with the lattice diagram for noncyclic group $U(n)$ can also be developed.

References

[1] Deitel, H. M. and Deitel, P. J. 2022. C++ *How to Program*. 4th Edition. United State America: Prentice Hall.

[2] Mohd Ali, N. M., Lim, D.S.F., Sarmin, N.H., Salleh, S. 2006. *A Visual Model For Computing Some Properties of U(n) and Zn*. Proceedings of the 2nd IMT-GT Regional Conference on Mathematics, Statistics and Applications. Universiti Sains Malaysia. 29-35.

[3] Mohd Ali, N. M., Noor Azhuan, N. A., Sarmin, N. H. and Johar, F. 2017. *The Computation of Some Properties of Addictive and Multiplicative Groups of Integers Modulo n Using C++ Programming*. Sains Humanika. 9(1-2): 57-63.

[4] Abd Rahman. 2019. *The Computation Software Model For Visualising Some Properties of Some Finite Abelian Groups and Dihedral Group*. Faculty of Science, Universiti Teknologi Malaysia.

[5] Stroustrup, B. 1986. *The C++ Programming Language.* Reading, MA: Addison-Wesley.

[6] Wussing, H. 2007. *The genesis of the abstract group concept: a contribution to the history of the origin of abstract group theory.* New York: Dover Publications.

[7] Gray, J. 2018. *A History of Abstract Algebra.* Springer Undergraduate Mathematics Series. 281-288.

[8] Sarmin, Mat Hassim, Mohd Ali. 2019. *Modern Algebra* (7th Edition). Department of Mathematical Sciences, Faculty of Science, Universiti Teknologi Malaysia.

[9] Gallian, J. 2012. *Contemporary abstract algebra*. USA: Nelson Education. 33

[10] Brodie, M. 2016. *Subgroup Lattices of Finite Cyclic Groups.* Wolfram Demonstration Project. online, retrieved 15th May 2024 from https://demonstrations.wolfram.com/SubgroupLatticesOfFiniteCyclicGroups/