

Patterns of Algorithms - Examples

Example 1 – Sequence calculation

Average of marks: We know the student's marks from a given subject. Let's calculate the average of marks.

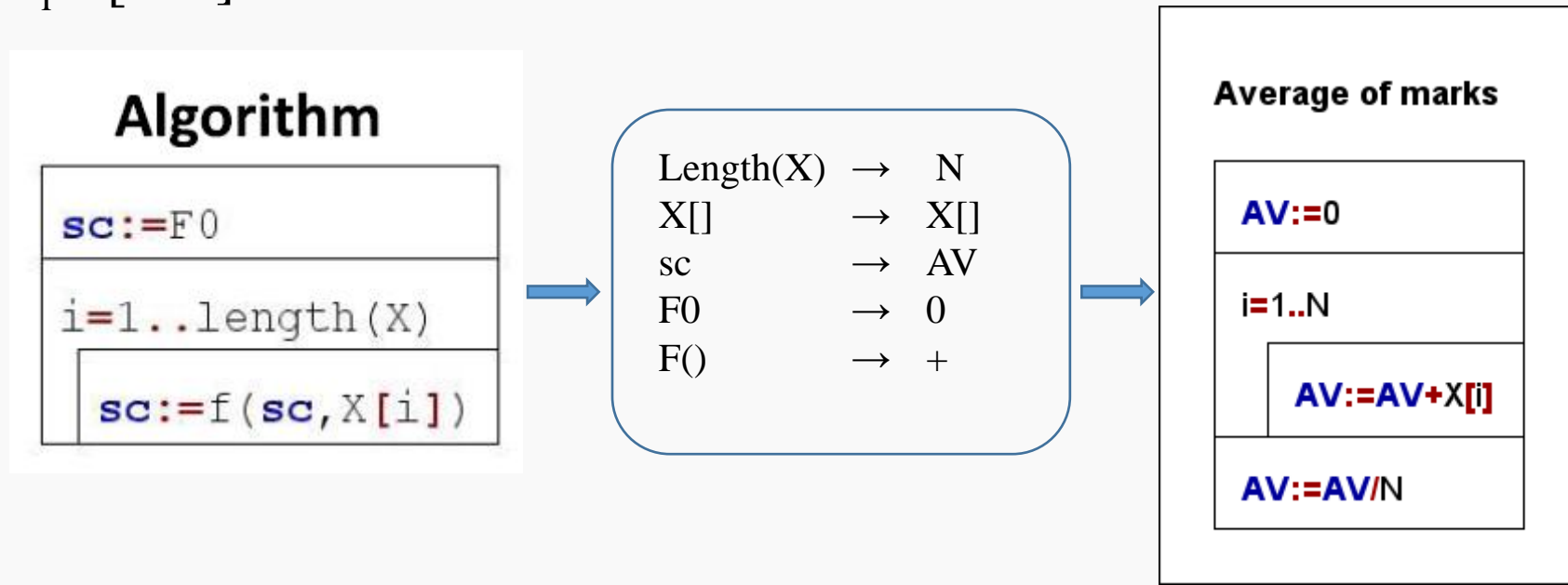
Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{N}^N$

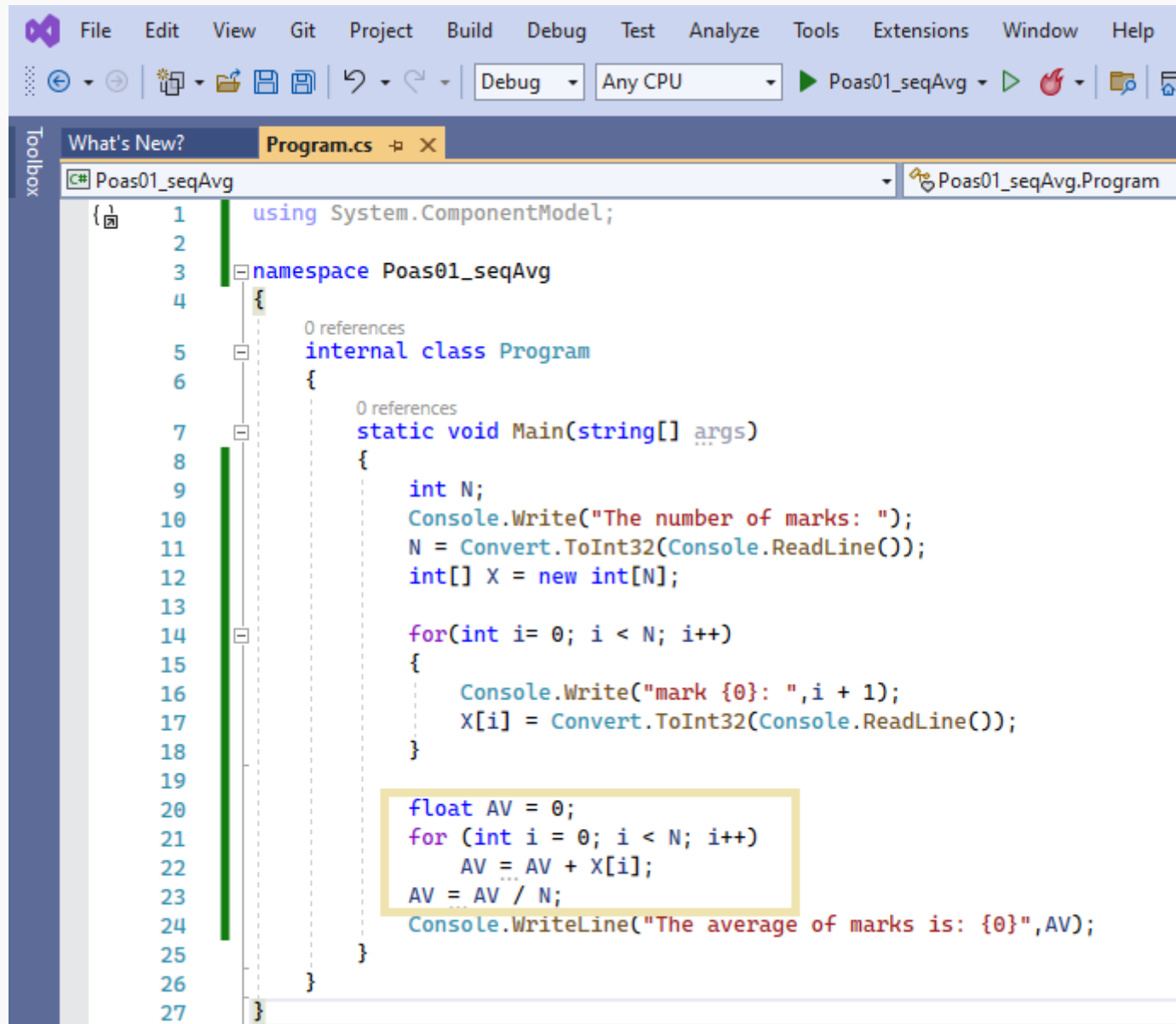
Output: $AV \in \mathbb{R}$

Precondition: $\forall i (1 \leq i \leq N) : X_i \in [1..5]$

Postcondition: $AV = \frac{\sum_{i=1}^N X_i}{N}$



Source code 1 – Sequence calculation – Average of marks



The image shows a screenshot of the Visual Studio IDE. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. Below the menu bar is a toolbar with various icons for navigation and development. The main editor window displays the source code for a C# program named 'Program.cs'. The code is for a console application that calculates the average of marks. The code is as follows:

```
1 using System.ComponentModel;
2
3 namespace Poas01_seqAvg
4 {
5     0 references
6     internal class Program
7     {
8         0 references
9         static void Main(string[] args)
10        {
11            int N;
12            Console.Write("The number of marks: ");
13            N = Convert.ToInt32(Console.ReadLine());
14            int[] X = new int[N];
15
16            for(int i= 0; i < N; i++)
17            {
18                Console.Write("mark {0}: ",i + 1);
19                X[i] = Convert.ToInt32(Console.ReadLine());
20            }
21
22            float AV = 0;
23            for (int i = 0; i < N; i++)
24                AV = AV + X[i];
25            AV = AV / N;
26            Console.WriteLine("The average of marks is: {0}",AV);
27        }
28    }
29 }
```

The code is written in C# and is for a console application. It starts with a using statement for System.ComponentModel. It then defines a namespace Poas01_seqAvg. Inside this namespace, there is an internal class Program. The Program class has a static void Main method that takes a string array args as a parameter. The Main method contains the following logic: it prompts the user for the number of marks (N), reads the input, and converts it to an integer. It then creates an array X of size N. It enters a loop that prompts the user for each mark (i+1), reads the input, and converts it to an integer, storing it in X[i]. After the loop, it calculates the average (AV) by summing all the marks in X and dividing by N. Finally, it prints the average to the console.

Example 2 – Sequence calculation

Product of a and b by addition: Suppose our computer knows only one operation, that is addition. Let's calculate the product of **a** and **b** by addition.

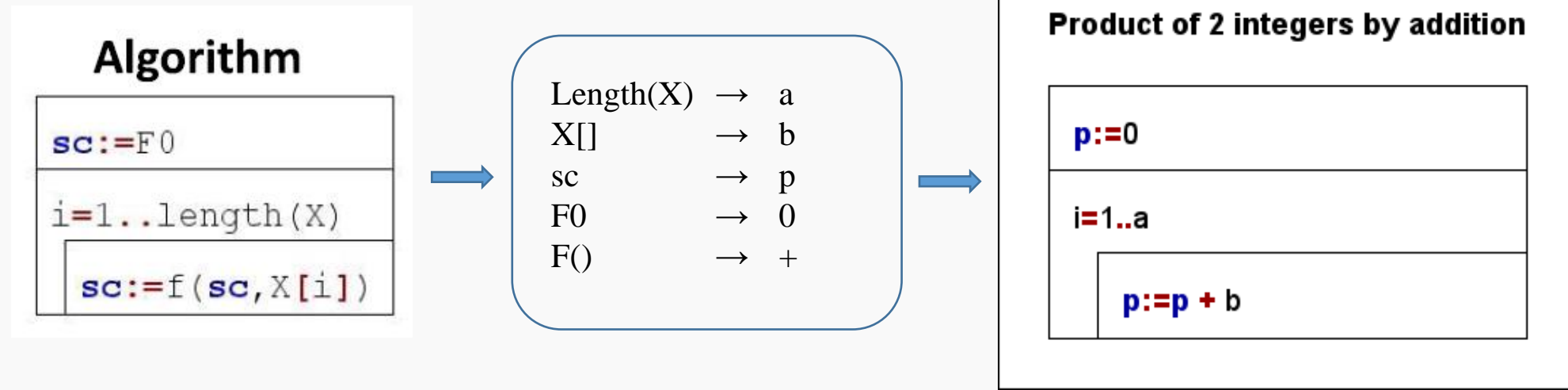
Specification:

Input: $a, b \in \mathbb{Z}$

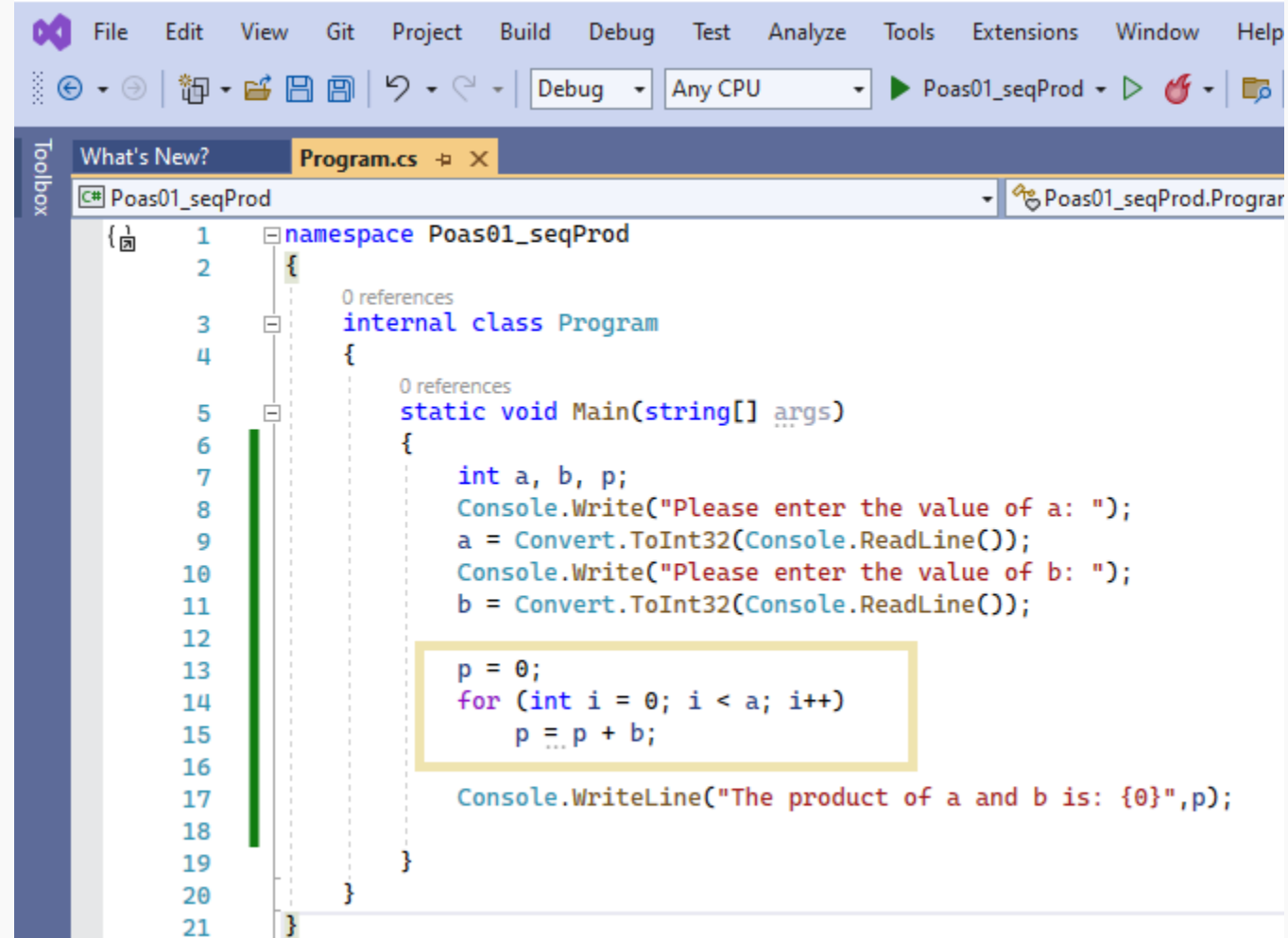
Output: $p \in \mathbb{Z}$

Precondition: $a, b \neq 0$

Postcondition: $p = \sum_{i=1}^a b$



Source code 2 – Sequence calculation – Product of a and b by addition



The image shows a screenshot of the Visual Studio IDE. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. Below the menu bar is a toolbar with icons for navigation and execution. The main window displays a C# file named Program.cs. The code defines a namespace Poas01_seqProd and an internal class Program. Inside the Program class, there is a static void Main method that takes an array of strings as input. The Main method prompts the user to enter the value of a and b, reads the input, and then calculates the product of a and b using a loop that adds b to a variable p. The final result is printed to the console.

```
1 namespace Poas01_seqProd
2 {
3     0 references
4     internal class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             int a, b, p;
10            Console.Write("Please enter the value of a: ");
11            a = Convert.ToInt32(Console.ReadLine());
12            Console.Write("Please enter the value of b: ");
13            b = Convert.ToInt32(Console.ReadLine());
14
15            p = 0;
16            for (int i = 0; i < a; i++)
17                p = p + b;
18
19            Console.WriteLine("The product of a and b is: {0}",p);
20        }
21    }
```

Example 3 – Counting

The number of even numbers divisible by 7: Let's count the number of even and divisible by 7 numbers from an array.

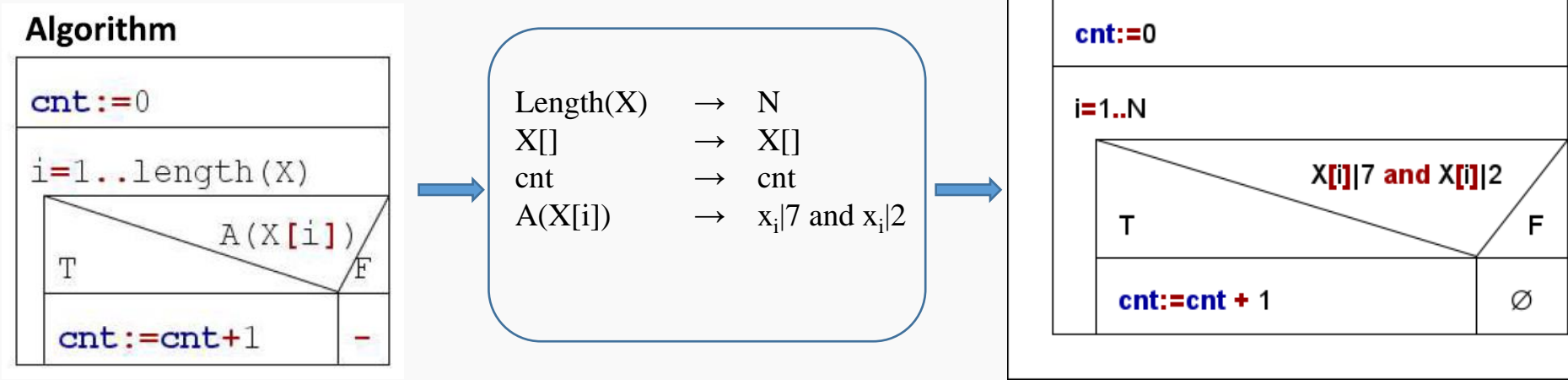
Specification:

Input: $N \in \mathbb{N}, X_{1..N} \in \mathbb{N}^N$
 $A: \mathbb{N} \rightarrow \mathbb{L}$ $A(x) := (x \mid 7 \text{ and } x \mid 2)$

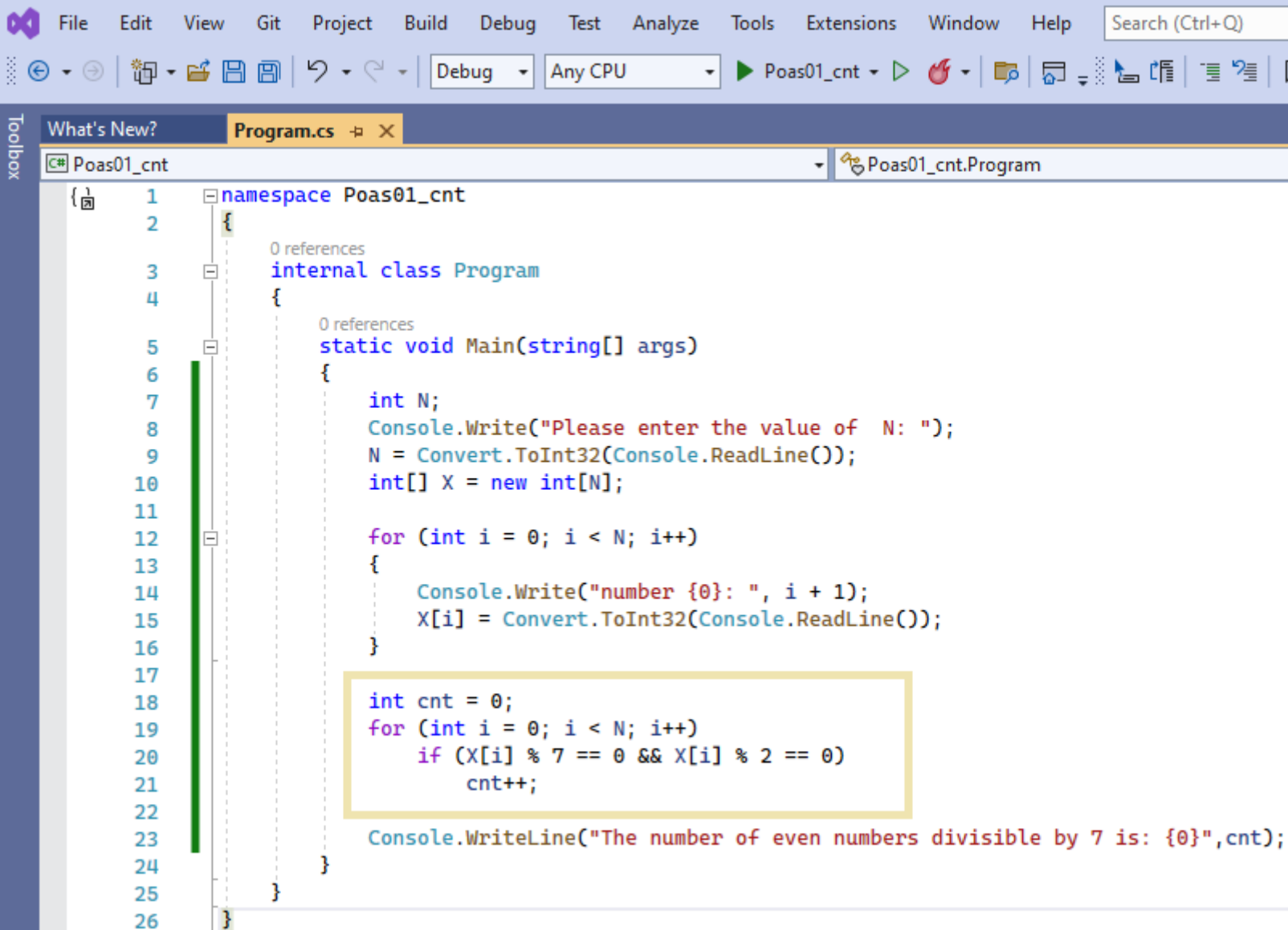
Output: $cnt \in \mathbb{N}$

Precondition:

Postcondition: $cnt = \sum_{i=1}^N 1_{A(X_i)}$



Source code 3 – Counting – The number of even numbers divisible by 7



```
1 namespace Poas01_cnt
2 {
3     0 references
4     internal class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             int N;
10            Console.Write("Please enter the value of N: ");
11            N = Convert.ToInt32(Console.ReadLine());
12            int[] X = new int[N];
13
14            for (int i = 0; i < N; i++)
15            {
16                Console.Write("number {0}: ", i + 1);
17                X[i] = Convert.ToInt32(Console.ReadLine());
18
19                int cnt = 0;
20                for (int i = 0; i < N; i++)
21                    if (X[i] % 7 == 0 && X[i] % 2 == 0)
22                        cnt++;
23
24                Console.WriteLine("The number of even numbers divisible by 7 is: {0}", cnt);
25            }
26        }
27    }
28 }
```

Example 4 – Maximum selection

The longest name: There is a list with the name of students. Let’s select the longest name from this list.

Specification:

Input: $N \in \mathbb{N}, X_{1..N} \in \mathbb{S}^N$

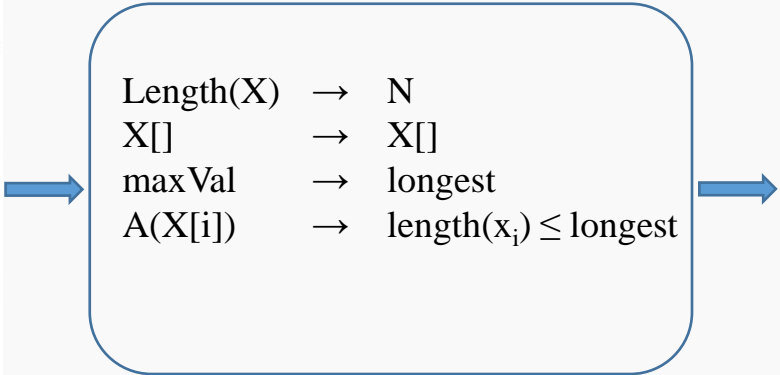
Output: $\text{longest} \in \mathbb{S}$

Precondition: $N > 0$ and $\exists i (1 \leq i \leq N) : \text{length}(X_i) > 0$

Postcondition: $\text{longest} \in X$ and $\exists i (1 \leq i \leq N) : \text{longest} = X_i$ and $\forall i (1 \leq i \leq N) : \text{longest} \geq \text{length}(X_i)$

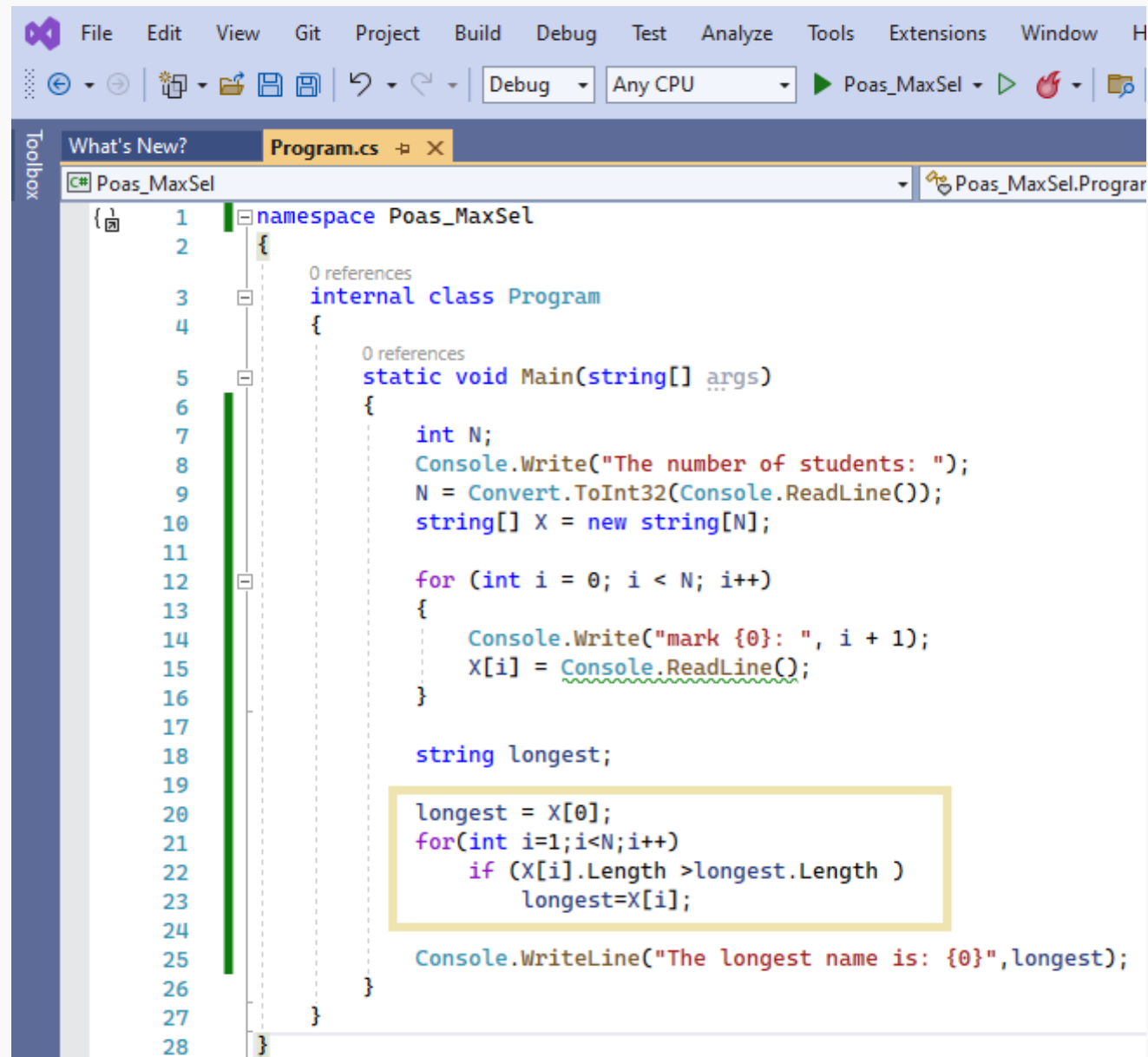
Algorithm

maxVal := X[1]	
i = 2 .. length(X)	
<div><div></div><div><div><div>X[i] > maxVal</div></div><div>T</div><div>F</div></div></div>	
maxVal := X[i]	-



The longest name	
longest := X[1]	
i = 2 .. N	
<div><div></div><div><div><div>length(X[i]) > length(longest)</div></div><div>T</div><div>F</div></div></div>	
longest := X[i]	∅

Source code 4 – Maximum selection – The longest name



The image shows a screenshot of the Visual Studio IDE. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. Below the menu bar is a toolbar with icons for navigation and execution. The main window displays a C# file named 'Program.cs' within a project called 'Poas_MaxSel'. The code defines a namespace 'Poas_MaxSel' containing an internal class 'Program'. Inside the 'Program' class, there is a static method 'Main' that takes an array of strings 'args' as input. The method reads an integer 'N' from the console, creates an array 'X' of strings of size 'N', and then iterates through the array to find the longest string. The longest string is stored in the variable 'longest', which is then printed to the console. The code is as follows:

```
1 namespace Poas_MaxSel
2 {
3     internal class Program
4     {
5         static void Main(string[] args)
6         {
7             int N;
8             Console.Write("The number of students: ");
9             N = Convert.ToInt32(Console.ReadLine());
10            string[] X = new string[N];
11
12            for (int i = 0; i < N; i++)
13            {
14                Console.Write("mark {0}: ", i + 1);
15                X[i] = Console.ReadLine();
16            }
17
18            string longest;
19
20            longest = X[0];
21            for(int i=1;i<N;i++)
22                if (X[i].Length > longest.Length )
23                    longest=X[i];
24
25            Console.WriteLine("The longest name is: {0}",longest);
26        }
27    }
28 }
```

Example 5 – Search

Square number: Let's looking for a square number within a sequence of numbers.

Specification: Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{N}^N$, $A: \mathbb{N} \rightarrow \mathbb{L}$

Output: $\text{Exists} \in \mathbb{L}$, $\text{Ind} \in \mathbb{N}$, $\text{Val} \in \mathbb{N}$

Precondition: $\forall i (1 \leq i \leq N) : X_i \geq 0$

Postcondition: $\text{Exists} = (\exists \text{ind} (1 \leq \text{ind} \leq N) : \text{sqrt}(X_{\text{ind}}) \text{ is Integer})$
and $\text{Exists} \rightarrow 1 \leq \text{ind} \leq N \text{ and } \text{sqrt}(X_{\text{ind}}) \text{ is Integer}$

Algorithm

$i := 1$	
$i \leq \text{length}(X)$ and not $A(X[i])$	
$i := i + 1$	
$\text{exists} := (i \leq \text{length}(X))$	
T	F
$\text{ind} := i$	$-$
$\text{value} := X[i]$	



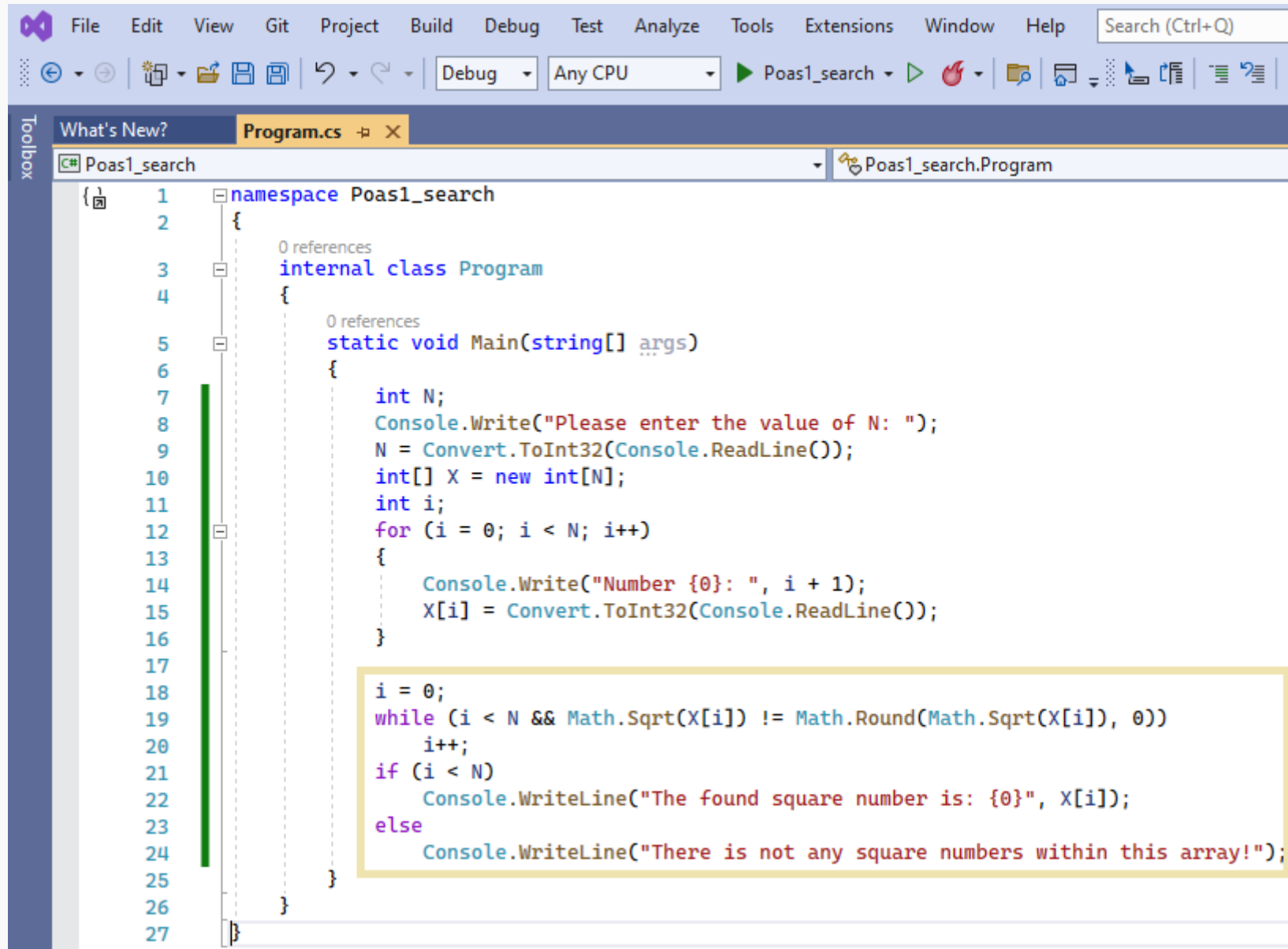
Length(X)	$\rightarrow N$
X[]	$\rightarrow X[]$
existx	$\rightarrow \text{Exists}$
$A(X[i])$	$\rightarrow \text{sqrt}(X[i]) = \text{int}(\text{sqrt}(X[i]))$



Search Square Number

$i := 1$	
$i \leq N$ and $\text{sqrt}(X[i])$ is not Integer	
$i := i + 1$	
$\text{Exist} := i \leq N$	
T	F
$\text{Ind} := i$	\emptyset
$\text{Val} := X[i]$	

Source code 5 – Search – Square number



The image shows a screenshot of the Visual Studio IDE. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. Below the menu bar is a toolbar with various icons for file operations, debugging, and development. The main editor window displays a C# file named Program.cs. The code defines a namespace Poas1_search and an internal class Program. The Program class has a static void Main method that takes an array of strings as input. Inside the Main method, it prompts the user to enter a value N, reads it from the console, and creates an array X of size N. It then iterates through the array X, checking if each element is a perfect square. If it is, it prints the square number; otherwise, it prints a message indicating that no square numbers were found within the array.

```
1 namespace Poas1_search
2 {
3     0 references
4     internal class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             int N;
10            Console.WriteLine("Please enter the value of N: ");
11            N = Convert.ToInt32(Console.ReadLine());
12            int[] X = new int[N];
13            int i;
14            for (i = 0; i < N; i++)
15            {
16                Console.WriteLine("Number {0}: ", i + 1);
17                X[i] = Convert.ToInt32(Console.ReadLine());
18            }
19            i = 0;
20            while (i < N && Math.Sqrt(X[i]) != Math.Round(Math.Sqrt(X[i]), 0))
21            {
22                i++;
23            }
24            if (i < N)
25            {
26                Console.WriteLine("The found square number is: {0}", X[i]);
27            }
28            else
29            {
30                Console.WriteLine("There is not any square numbers within this array!");
31            }
32        }
33    }
34 }
```

Example 6 – Decision

Divisible by 3: Let's make a decision is there any number that is divisible by 3 within a sequence of numbers.

Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{N}^N$, $A: \mathbb{N} \rightarrow \mathbb{L}$, $A(x) := (x \mid 3)$

Output: $\text{Exists} \in \mathbb{L}$

Precondition: –

Postcondition: $\text{Exists} = \exists i (1 \leq i \leq N): X_i \mid 3$

Algorithm

i := 1

i ≤ length(X) **and not** A(X[**i**])

i := **i** + 1

exists := (**i** ≤ length(X))



Length(X) → N
X[] → X[]
existx → Exists
A(X[i]) → Xi|3



Divisible by 3

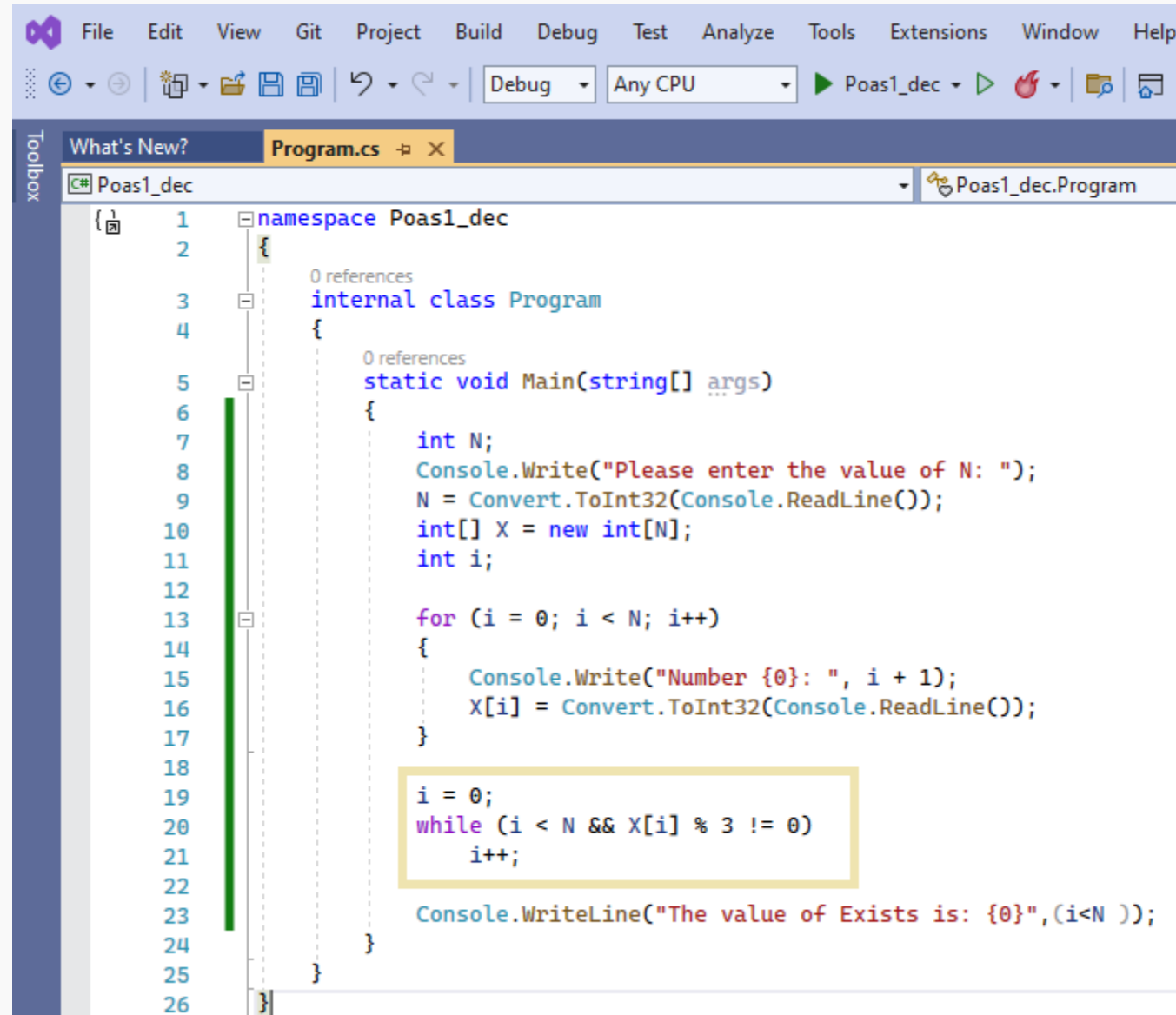
i := 1

i ≤ N **and not** (X[**i**] | 3)

i := **i** + 1

Exists := (**i** ≤ N)

Source code 6 – Decision – Divisible by 3



```
1 namespace Poas1_dec
2 {
3     0 references
4     internal class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             int N;
10            Console.Write("Please enter the value of N: ");
11            N = Convert.ToInt32(Console.ReadLine());
12            int[] X = new int[N];
13            int i;
14
15            for (i = 0; i < N; i++)
16            {
17                Console.Write("Number {0}: ", i + 1);
18                X[i] = Convert.ToInt32(Console.ReadLine());
19            }
20
21            i = 0;
22            while (i < N && X[i] % 3 != 0)
23            {
24                i++;
25            }
26            Console.WriteLine("The value of Exists is: {0}",(i<N ));
27        }
28    }
29 }
```

Example 7 – Selection

Four digits number: Let's select the first four digits number from a sequence of numbers.

Specification:

Input: $N \in \mathbb{N}$, $X_{1..N} \in \mathbb{N}^N$, $A: \mathbb{N} \rightarrow \mathbb{L}$

Output: $\text{Index} \in \mathbb{N}$, $\text{Value} \in \mathbb{N}$

Precondition: $N > 0$ and $\exists i (1 \leq i \leq N): A(X_i)$

Postcondition: $1 \leq \text{Index} \leq N$ and $1000 \leq X_{\text{Index}} \leq 9999$

Algorithm

$i := 1$
$\text{not } A(X[i])$
$i := i + 1$
$\text{ind} := i$
$\text{val} := X[i]$



$\text{Length}(X) \rightarrow N$
$X[] \rightarrow X[]$
$\text{ind} \rightarrow \text{Index}$
$A(X[i]) \rightarrow 999 < X[i] < 10\,000$



Four digits number

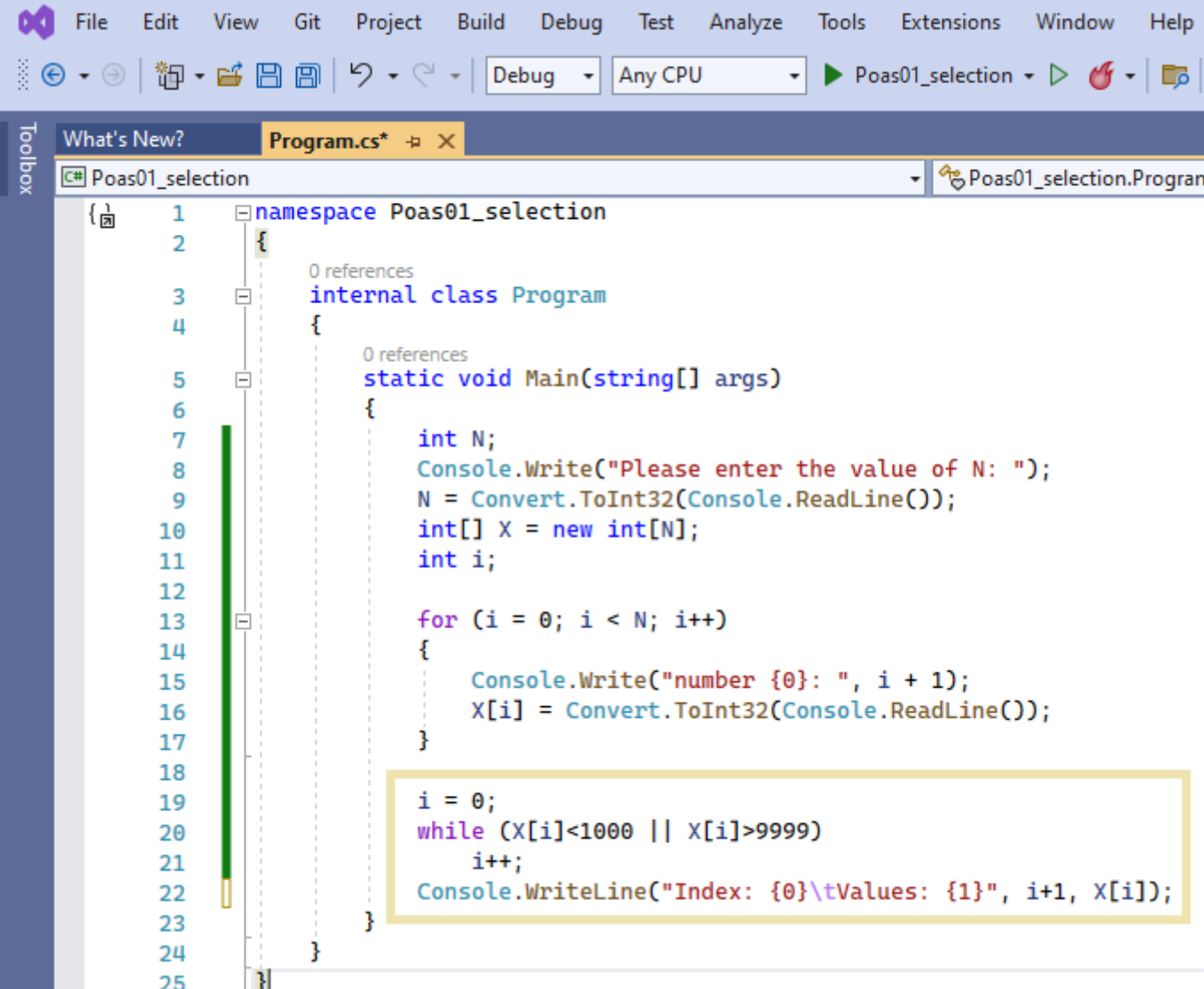
$i := 1$
$\text{not } (X[i] > 999 \text{ and } X[i] < 10000)$
$i := i + 1$
$\text{Index} := i$
$\text{Value} := X[i]$



Four digits number

$i := 1$
$X[i] < 1000 \text{ and } X[i] > 9999$
$i := i + 1$
$\text{Index} := i$
$\text{Value} := X[i]$

Source code 7 – Selection – Four digits number



```
1 namespace Poas01_selection
2 {
3     0 references
4     internal class Program
5     {
6         0 references
7         static void Main(string[] args)
8         {
9             int N;
10            Console.Write("Please enter the value of N: ");
11            N = Convert.ToInt32(Console.ReadLine());
12            int[] X = new int[N];
13            int i;
14
15            for (i = 0; i < N; i++)
16            {
17                Console.Write("number {0}: ", i + 1);
18                X[i] = Convert.ToInt32(Console.ReadLine());
19            }
20
21            i = 0;
22            while (X[i]<1000 || X[i]>9999)
23                i++;
24            Console.WriteLine("Index: {0}\tValues: {1}", i+1, X[i]);
25        }
26    }
27 }
```