Webology (ISSN: 1735-188X)
Volume 19, Number 5, 2022

Engineering Problem Solving Using Code Patterns By Means Of Rectangular Figures

Albert Miyer Suarez Castrillón[1], Rafael Bolívar León[2], Sir-Alexci Suarez Castrillon[3]

[1, 2] Faculty of Engineering and Architecture, GIMUP Research Group. Universidad de Pamplona, Colombia.

[3]Faculty of Engineering, GRUCITE Research Group, University Francisco de Paula Santander Ocaña, Colombia

ABSTRACT:
This article presents a code pattern based on using a rectangular or square figure, which can also be used by means of matrices. The objective is to use the code pattern to solve and find the solution to different problems, if you adapt the answers to a space in the rectangular figure, this way you can find the solution faster by having a logical coding pattern already defined. The idea is that many exercises always need 2 repetitive structures, and keep their same pattern if they need to be nested. Although it can be implemented with arrays, it is intended for students who are new to programming and need a tool to find the solution without advanced knowledge.

Keywords: Programming, Pseint, code pattern, rectangular shapes.

1. INTRODUCTION

Learning programming can help to solve problems, where several phases must be implemented, from analysis to program execution (Jimenez, 2022). The design and development of an algorithm is the initial basis for a good program, it is necessary before writing the first line of a program to have made the algorithm (Fonden-Calzadilla et al., 2018). An algorithm is the solution to a problem through defined steps to then encode it in a programming language; within the tools we can mention visual programs such as Kodo (Suarez Castillón & Soto Arévalo, 2015) or Pseint a program to create algorithms that is used in Latin America, and helps algorithmic thinking (Alanoca Gutierrez, 2016; Evangelista & Novara, 2014); The important thing is that you can program without knowing a programming language and see the result in execution.

The logic for programming in an initial course can have different nuances, from videos (Castrillon et al., 2021; Suarez et al., 2020) to extensive databases with solved exercises, and methodologies with learning platforms. Programming is a subject that is taught in almost all areas of knowledge, where the student faces a complex task, because not only to understand the concepts and structures of languages, but also to find solutions that usually end in the creation of a mathematical equation. Programming logic becomes a difficult barrier to overcome at the beginning, which may cause the student to withdraw from a basic programming course (Pimentel et al., 2012, 2022; Sãnchez et al., 2020).

Games are a very creative way to teach programming logic (Pereira et al., 2019), creating virtual worlds in 3D or through repetition cards (Vosinakis et al., 2018), one of the most popular tools is Scratch, which is an open environment to learn to program through blocks (Vázquez-Cano & Delgado, 2015), and it is taught from basic and high school cycles in schools. The experiences and results have been quite good (Durango-Warnes & Ravelo-Méndez, 2020; Narváez-Díaz & López-Martínez, 2021).

However, when solving real life problems, it always creates a complexity, which must be overcome or at least have a guide that allows to focus and find the way to solve the problem and then code in a programming language. The objective of this research is to present a code pattern which is based on rectangular or square figures that would serve the initial programmer in the solution of numerous programming problems, using the pattern and making small changes that adapt to the solution of the problem more quickly if it adapts to the figure, repetitive structures such as while, for and do-while can be used, using the Pseint algorithm language.

2. METHODOLOGY

To solve many problems and perform the solution with a programming language must incorporate repetitive structures, the drawback is generated at the beginning of the programming course, due to the little knowledge of the student, which is why it is convenient to create templates or patterns that help solve the problem faster and more efficiently, and concludes with a functional program. To do this it is convenient to think of a square or rectangle divided by cells as if it were a vector or matrix. The structure is used while rows are traversed with the first while and columns with the second while nested in the first. Each while must have a different counter, and the process is traversed as shown in Figure 1. Adding the counters is as shown in Figure 2.

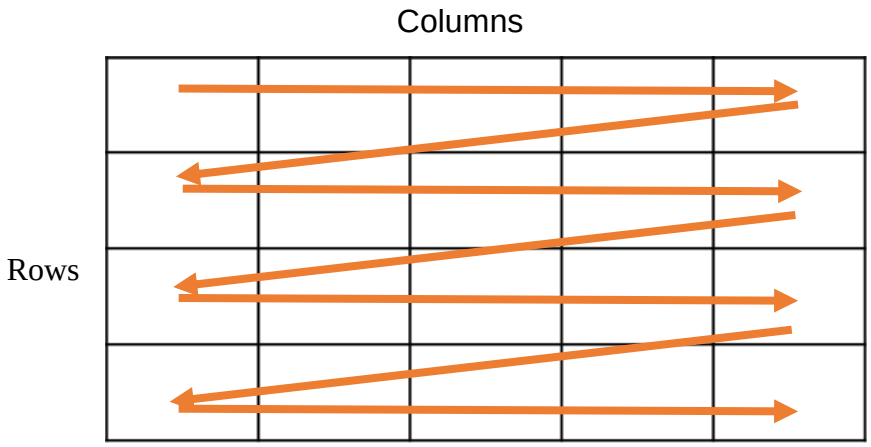

Figure 1. Data path direction

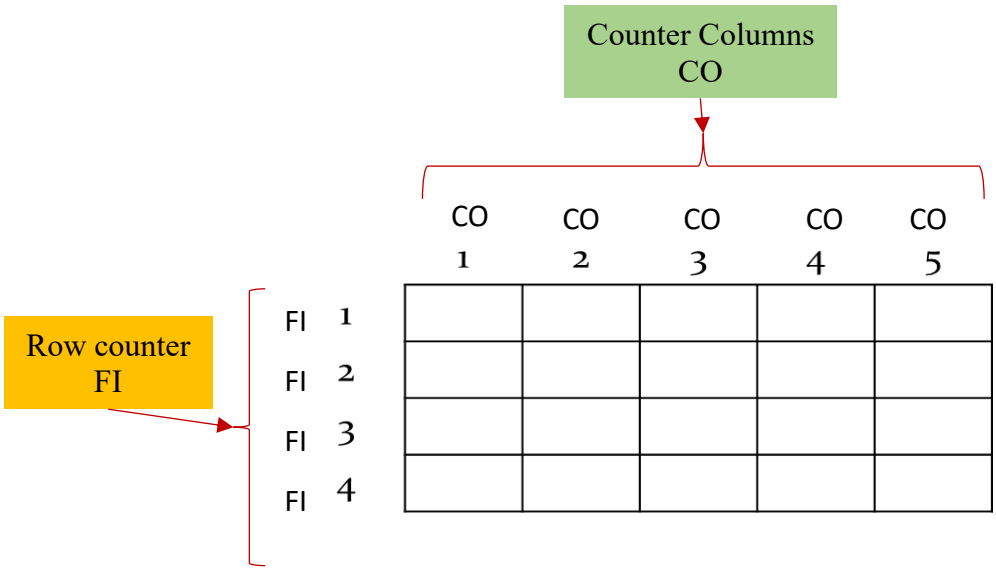

Figure 2. Adding counter to rows and columns

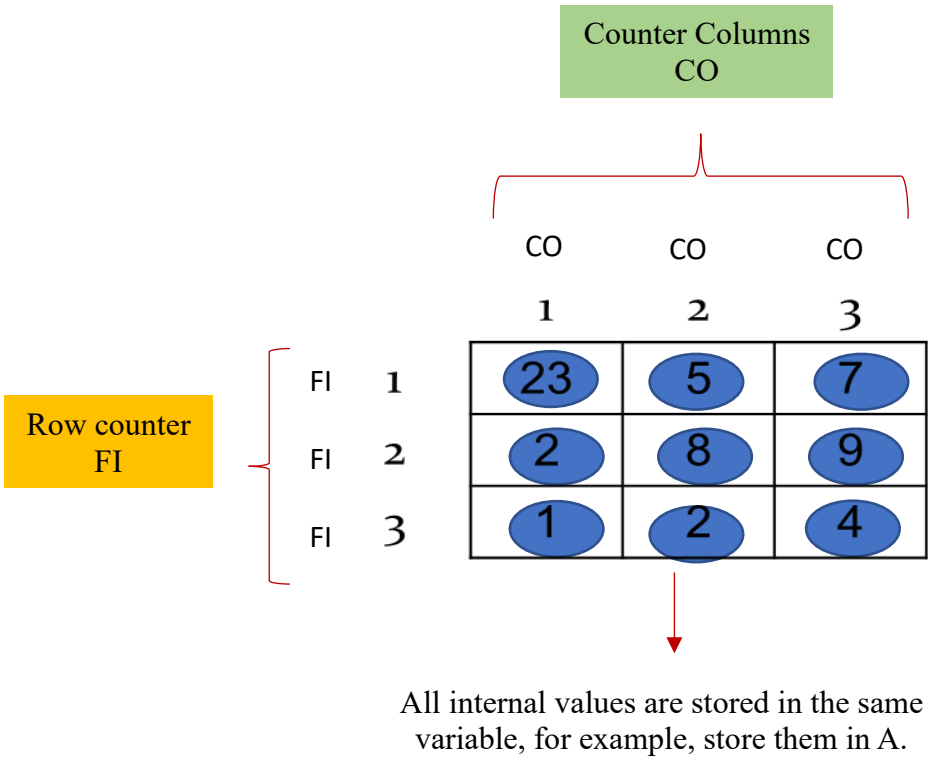

Figure 3. Adding values to each position

```
Algoritmo sin_titulo
    Definir Fi, Co, A como entero;
    Fi=1;
    mientras Fi≤3 Hacer
        Co=1;
        mientras Co≤3 Hacer
            Escribir "Ingrese valor Fila ",Fi, " Columna ", CO;
            Leer A;
            Co=Co+1;
        FinMientras
        Fi=Fi+1;
    FinMientras
FinAlgoritmo
```

Figure 4. Pattern while originating with squares.

The above pattern serves to solve innumerable problems (Figure 4), and only the exercise has to try to bring a rectangle as shown in Figure 5 and apply the pattern.

	1	2	3	4	5	6	7	8	9	10
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										

Figure 5. Rectangle or square where the solution is to be taken

3. RESULTS

Different problems are presented, which must be taken to the rectangle or square to apply the pattern and proceed to carry out the program. For example, if you have to draw the following figure 6, in the form of a triangle with asterisks, place each asterisk in a position of the rectangle and apply the pattern as in figure 7.

*

**

***

****

*****

Implemented in the rectangle

	1	2	3	4	5
1	*				
2	*	*			
3	*	*	*		

4	*	*	*	*	
5	*	*	*	*	*

Figure 6. Triangle taken to rectangle

```
1   Algoritmo sin_titulo
2       definir Fi, Co como entero;
3       Fi=1;
4       mientras Fi≤5 Hacer
5           Co=1;
6           mientras Co≤Fi Hacer
7               Escribir "*" Sin Saltar;
8               Co=Co+1;
9           FinMientras
10          Escribir "";
11          Fi=Fi+1;
12      FinMientras
13  FinAlgoritmo
```

Figure 7. Application of the pattern for the triangle problema

The same pattern can be used to realize the shape of a tree or the Pascal binomial, where the asterisks are replaced by values (Figure 8).

```
*
***
*****
*******
*********
***********
```

	1	2	3	4	5	6	7	8	9	10	11
1						*					
2					*	*	*				
3				*	*	*	*	*			
4			*	*	*	*	*	*	*		
5		*	*	*	*	*	*	*	*	*	
6	*	*	*	*	*	*	*	*	*	*	*

Figure 8. Tree shape when taken to the rectangle.

It can also be used in the solution of a multiplication table, for example if you need to show the table of the number 3 (Table 1). When taken to the rectangle it looks like Table 2. Where the

rows represent the first column from 1 to 10, and the table number to be printed is the column, and the result of multiplication is placed in each space according to the position of rows and columns. Figure 9 shows the implementation of the code by means of the rectangular patterns.

Table 1. Multiplication table of 3.

Files		Columns		Resulted
1	*	3	=	3
2	*	3	=	6
3	*	3	=	9
4	*	3	=	12
5	*	3	=	15
6	*	3	=	18
7	*	3	=	21
8	*	3	=	24
9	*	3	=	27
10	*	3	=	30

Table 2. Multiplication table of 3 with solution in the rectangle

	1	2	3	4	5	6	7	8	9	10
1			3							
2			6							
3			9							
4			12							
5			15							
6			18							
7			21							
8			24							
9			27							
10			30							

```
1   Algoritmo sin_titulo
2       definir Fi, Co,T,R como entero;
3       Escribir "Tabla "; Leer T;
4       Fi=1;
5       Co=1;
6       mientras Fi≤1 Hacer
7           mientras Co≤10 Hacer
8               R=Co*T;
9               Escribir Fi,"*",T,"=",R;
10              Co=Co+1;
11          FinMientras
12          Fi=Fi+1;
13      FinMientras
14  FinAlgoritmo
```

Figure 9. Implementation of the pattern for the keyboard entered table code

If all the multiplication tables are needed, up to 10; as it appears in the teaching notebooks in the basic cycle, it needs to be shown as table 3, when taken to the table the results are shown as table 4, and when using the code pattern the solution is the same as Figure 10.

Table 3. All multiplication tables

1	*	1	=	1		1	*	2	=	2		1	*	3	=	3		.	.	.	.	.		1	*	10	=	10
2	*	1	=	2		2	*	2	=	4		2	*	3	=	6		.	.	.	.	.		2	*	10	=	20
3	*	1	=	3		3	*	2	=	6		3	*	3	=	9		.	.	.	.	.		3	*	10	=	30
4	*	1	=	4		4	*	2	=	8		4	*	3	=	12		.	.	.	.	.		4	*	10	=	40
5	*	1	=	5		5	*	2	=	10		5	*	3	=	15		.	.	.	.	.		5	*	10	=	50
6	*	1	=	6		6	*	2	=	12		6	*	3	=	18		.	.	.	.	.		6	*	10	=	60
7	*	1	=	7		7	*	2	=	14		7	*	3	=	21		.	.	.	.	.		7	*	10	=	70
8	*	1	=	8		8	*	2	=	16		8	*	3	=	24		.	.	.	.	.		8	*	10	=	80
9	*	1	=	9		9	*	2	=	18		9	*	3	=	27		.	.	.	.	.		9	*	10	=	90
10	*	1	=	10		10	*	2	=	20		10	*	3	=	30		.	.	.	.	.		10	*	10	=	100

Table 4. All the multiplication tables in the rectangle

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

```
1  Algoritmo sin_titulo
2      definir Fi, Co,R como entero;
3      Fi=1;
4      Co=1;
5      mientras Fi≤10 Hacer
6          Co=1;
7          mientras Co≤10 Hacer
8              R=Fi*Co;
9              Escribir Fi,"*",Co,"=",R;
10             Co=Co+1;
11         FinMientras
12         Fi=Fi+1;
13     FinMientras
14 FinAlgoritmo
```

Figure 10. Implementation of pattern for all multiplication tables

If we want to find the prime numbers from 2 to 20, we need the number to be divisible exactly by unity and by itself, for this we need to find the remainder, and if we count the zeros of the remainders, the result will give in the prime number if the remainder is once 0 (Table 5 and Figure 11).

Table 5. Residuals for finding prime numbers

	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Cantidad ceros	Primo
2	0														1	Yes
3	1	0													1	Yes
4	0	1	0												2	No
5	1	2	1	0											1	Yes
6	0	0	2	1	0										3	No
7	1	1	3	2	1	0									1	Yes
8	0	2	0	3	2	1	0								3	No
9	1	0	1	4	3	2	1	0							2	No

10	0	3	2	0	4	3	2	1	0						3	No
11	1	2	3	6	5	4	3	2	1	0					1	Yes
12	0	0	0	2	0	5	4	3	2	1	0				4	No
13	1	1	1	3	1	6	5	4	3	2	1	0			1	Yes
14	0	2	2	4	2	0	6	5	4	3	2	1	0		3	No
15	1	0	3	0	3	8	7	6	5	4	3	2	1	0	3	No

```
1  Algoritmo sin_titulo
2      definir Fi, Co,R,P como entero;
3      Fi=2;
4      Co=2;
5      mientras Fi≤15 Hacer
6          Co=2;
7          P=0;
8          mientras Co≤Fi Hacer
9              R=Fi mod Co;
10             si R==0 Entonces
11                 P=P+1;
12             SiNo
13
14             FinSi
15             Co=Co+1;
16         FinMientras
17         Si P==1 Entonces
18             Escribir "Es Primo",Fi;
19         FinSi
20         Fi=Fi+1;
21     FinMientras
22 FinAlgoritmo
```

Figure 11. Implementation of the pattern for all prime numbers

Another exercise can be the factorial of a number. Develop a program to find the factorial of a number entered by keyboard.

For example:

The factorial of 2 is 2 comes out of multiplying 1*2.

The factorial of 3 is 6 comes out of multiplying 1*2*3

The factorial of 4 is 24 comes out of multiplying 1*2*3*4

The factorial of 5 is 120 comes out of multiplying 1*2*3*4*5

If we want to find the factorial of 4 and take them to the rectangle, it would look like Table 6.

Table 6. Factorial in the rectangle

Number	1	2	3	4	Result
1					
2					
3					
4	1	2	6	24	24
…					
…					

Very many exercises can use the same rectangular pattern for the solution by means of 2 repetitive structures such as while, for or Do while.

3. CONCLUSIONS

The understanding of programming logic is complex for students who start in a programming course, and the solution to different problems is presented as a difficult moment, if you still do not have accurate knowledge. Usually many problems can be solved by taking the results to a rectangular or square figure and using a coding pattern with two nested repetitive structures, although it seems to behave like a matrix, it is not necessary for the student to have knowledge of matrices and vectors, since it can be used from the moment you start with the repetitive structures, and can help to find the right solution, just by having to implement the coding pattern.

REFERENCES

Alanoca Gutierrez, J. (2016). Pensamiento Algorítmico en la Matemática de la Enseñanza Básica. Revista Investigación y Tecnología, 01.

Castrillon, A. S., Castrillon, A. M. S., & Gegen, L. K. H. (2021). Design of Virtual Modules for the Development of Flipped Classroom in Programming.

Durango-Warnes, C., & Ravelo-Méndez, R. E. (2020). Beneficios del programa Scratch para potenciar el aprendizaje significativo de las Matemáticas en tercero de primaria *. Trilogía Ciencia Tecnología Sociedad, 12(23). https://www.redalyc.org/journal/5343/534368694008/html/

Evangelista, F. H., & Novara, P. J. (2014). Intérprete para probar un programa escrito en pseudocódigo. Industrial Data, 17(1), Art. 1. https://doi.org/10.15381/idata.v17i1.12039

Fonden-Calzadilla, J. C., Stuart-Cárdenas, M. L., & Rodríguez-Matos, L. (2018). La algoritmización: Requisito necesario para la solución de problemas con el empleo de un lenguaje de programación. Luz, 17(3), 30-43.

Jiménez, C. J. T. (2022). 3.1 Resolución de Problemas | 2016374 Programación en Lenguajes Estadísticos. https://bookdown.org/cjtorresj/ple/resoluci%C3%B3n-de-problemas.html

Narváez-Díaz, L. E., & López-Martínez, R. (2021). Creación de Videojuegos como Estrategia Educativa en Algoritmia. Revista Tecnológica-Educativa Docentes 2.0, 12(1), 22-30. https://doi.org/10.37843/rted.v1i1.219

Pereira, D. S., Santos, F. E. dos, & Lima, J. V. de. (2019). Educational Games in the Construction of Knowledge in Programming Logic. International Journal for Innovation Education and Research, 7(2), Art. 2. https://doi.org/10.31686/ijier.vol7.iss2.1324

Pimentel, J. J. A., García, O. S. N., González, R. S., & López, G. A. (2012). Software para la enseñanza-aprendizaje de algoritmos estructurados. Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología, 8, Art. 8. https://doi.org/10.24215/18509959.0.p

Pimentel, J. J. A., González, R. S., García, O. S. N., & Ibarra, S. P. C. (2022). Compilador e intérprete en línea de diagramas de flujo con fines didácticos. Revista de Investigación en Tecnologías de la Información, 10(20), Art. 20.

Sãnchez, M., Bahamondez, E. V., & de Clunie, G. T. (2020). Use of PSeInt in teaching programming: A case study. Proceedings of the 10th Euro-American Conference on Telematics and Information Systems, 1-5.

Suarez Castillón, S. A., & Soto Arévalo, F. S. (2015). Evaluación cualitativa de la utilización del lenguaje de programación visual kodu en niños de educación básica. Tecnura, 19(46), 37-48.

Suarez, S. A., Suarez, A. M., & Rincon, I. K. (2020). El nuevo coronavirus y la incidencia de videos modelizadores en la enseñanza de la lógica en programación. Revista ESPACIOS, 41(42). https://www.revistaespacios.com/a20v41n42/20414213.html

Vázquez-Cano, E., & Delgado, D. (2015). La creación de videojuegos con Scratch en Educación Secundaria. Communication Papers, 4, 63. https://doi.org/10.33115/udg_bib/cp.v4i06.22083

Vosinakis, S., Anastassakis, G., & Koutsabasis, P. (2018). Teaching and learning logic programming in virtual worlds using interactive microworld representations. British Journal of Educational Technology, 49(1), 30-44. https://doi.org/10.1111/bjet.12531