LENGUAJE
C

PROF. YULITH VANESSA ALTAMIRANO FLORES

REPORTE DE PRACTICA #4
FUNCIONES CON RETORNO Y SIN RETORNO

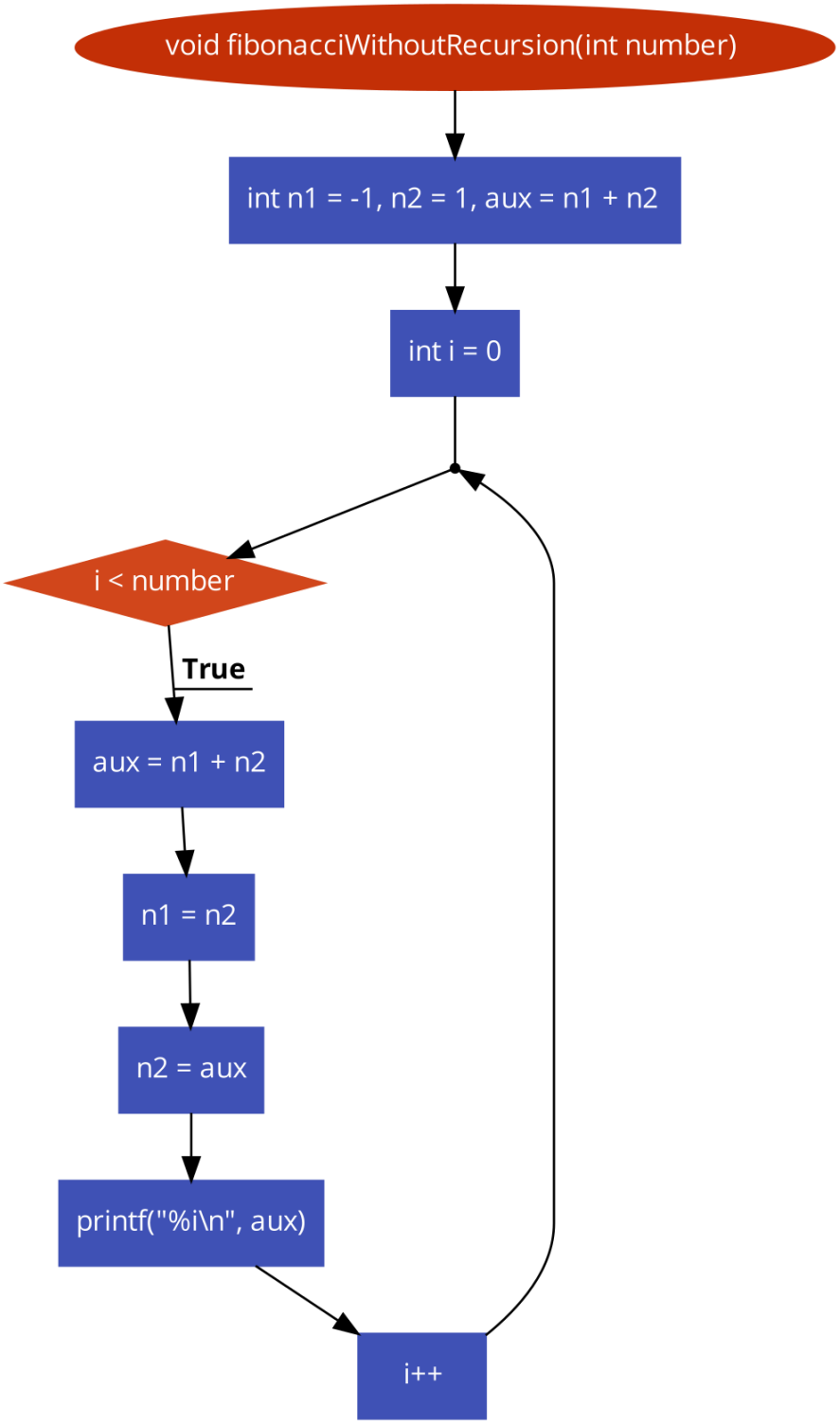
KEVIN ALEJANDRO GONZALEZ TORRES
GRUPO 932

REPOSITORIO

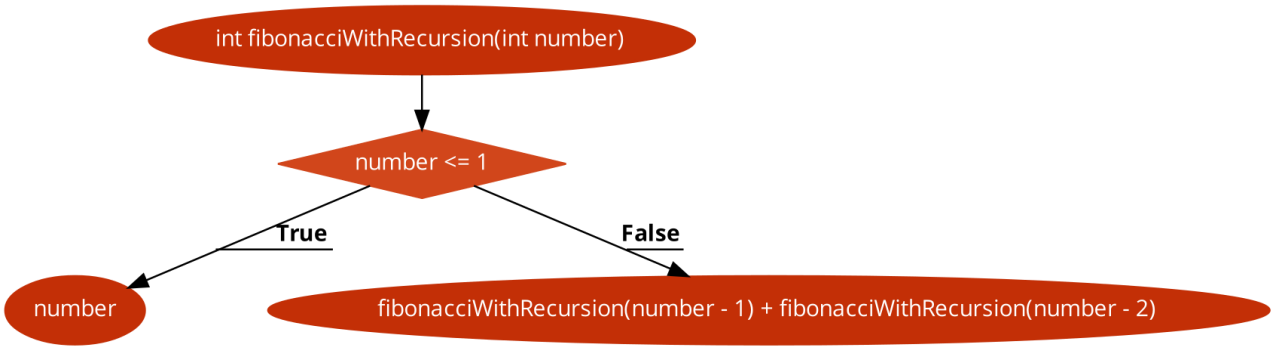
https://github.com/keevin-21/KAGT_Lenguaje_C_932

DIAGRAMA DE FLUJO

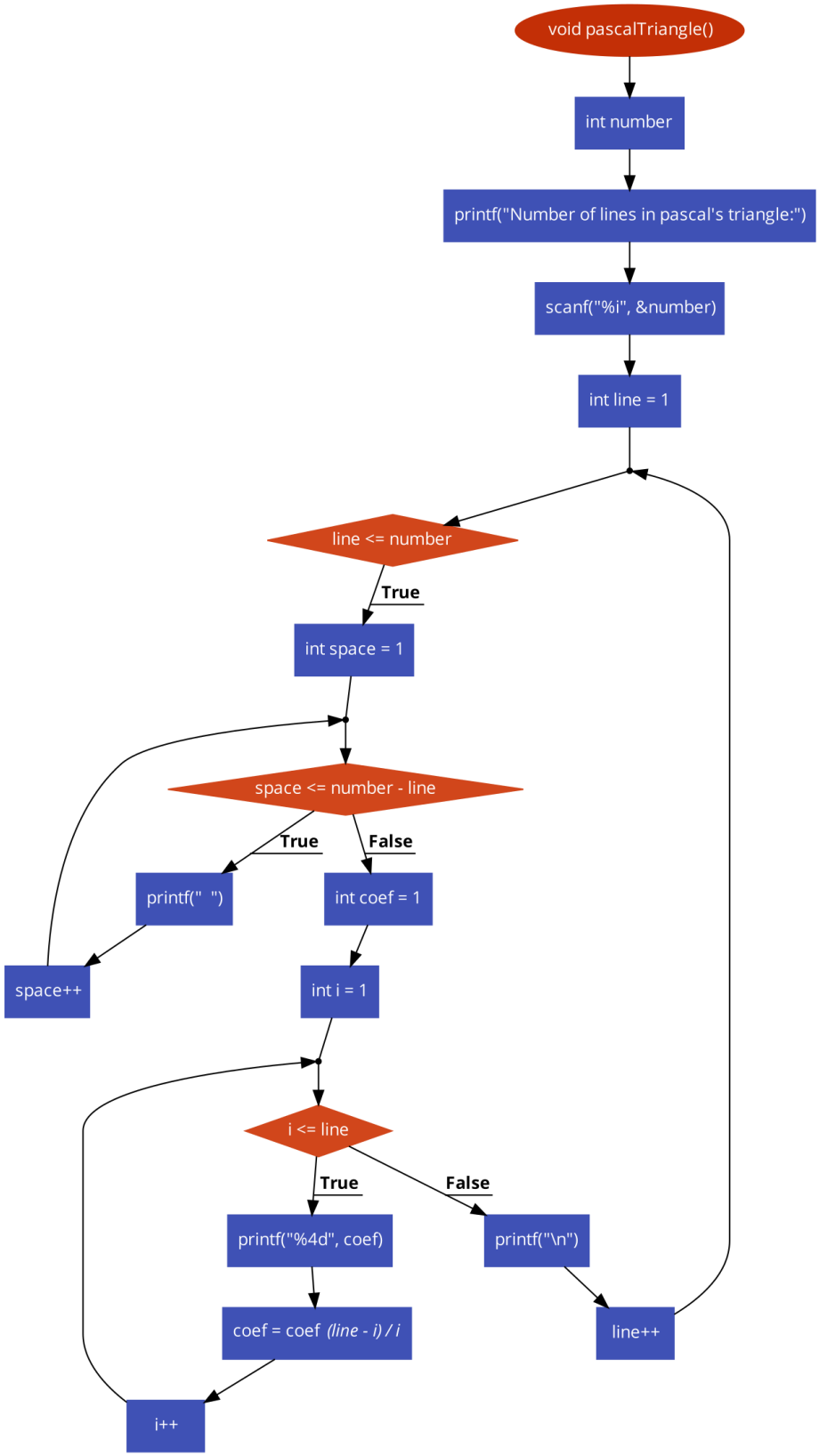
Actividad 1:



Actividad 2:


int fibonacciWithRecursion(int number)
number <= 1
True
False
number
fibonacciWithRecursion(number - 1) + fibonacciWithRecursion(number - 2)

Actividad 3:


void pascalTriangle()
int number
printf("Number of lines in pascal's triangle:")
scanf("%i", &number)
int line = 1
line <= number
True
int space = 1
space <= number - line
True
False
printf(" ")
int coef = 1
space++
int i = 1
i <= line
True
False
printf("%4d", coef)
printf("\n")
coef = coef (line - i) / i
line++
i++

FIBONACCI_SIN_RECURSION()

{

Variables:

```
94      int n1 = -1, n2 = 1, aux = n1 + n2;
```

Proceso principal:

```
96      // Calculate and print the Fibonacci sequence
97      for (int i = 0; i < number; i++)
98      {
99          aux = n1 + n2;
100         n1 = n2;
101         n2 = aux;
102         printf("%i\n", aux);
103     }
```

Variables:

- n1: Un entero inicializado en -1, se usa para almacenar el valor anterior en la secuencia de Fibonacci.
- n2: Un entero inicializado en 1, se usa para almacenar el valor actual en la secuencia de Fibonacci.
- aux: Un entero que se utiliza como una variable auxiliar para calcular el siguiente valor en la secuencia.

Proceso Principal:

- Esta función calcula y muestra la secuencia de Fibonacci sin utilizar recursión.
- Utiliza un bucle for para generar los números de Fibonacci.
- En cada iteración, se calcula aux como la suma de n1 y n2.
- Luego, n1 se actualiza con el valor de n2, y n2 se actualiza con el valor de aux.
- El valor calculado se imprime en cada iteración.

}

FIBONACCI_CON_RECURSION()

{

Proceso principal:

```
106  // Function to calculate the Fibonacci sequence with recursion
107  int fibonacciWithRecursion(int number)
108  {
109      if (number <= 1)
110      {
111          return number;
112      }
113      else
114      {
115          return fibonacciWithRecursion(number - 1) + fibonacciWithRecursion(number - 2);
116      }
117  }
```

Variables:

- number: Un entero que representa el índice del número de Fibonacci que se desea calcular.
- result: Un entero que almacena el resultado del cálculo del número de Fibonacci.

Proceso Principal:

- Esta función calcula el número de Fibonacci en la posición number utilizando recursión.
- Si number es menor o igual a 1, devuelve number (caso base).
- De lo contrario, llama recursivamente a la función fibonacciWithRecursion dos veces para calcular los números de Fibonacci anteriores y luego suma esos valores.
- Finalmente, devuelve el resultado.

}

TRIANGULO_PASCAL()

{

Variables:

```
122     int number;
```

Proceso Principal:

```
126     for (int line = 1; line <= number; line++)
127     {
128         // Print leading spaces for formatting
129         for (int space = 1; space <= number - line; space++)
130             printf("  ");
131 
132         int coef = 1;
133         for (int i = 1; i <= line; i++)
134         {
135             // Print coefficients with formatting
136             printf("%4d", coef);
137             coef = coef * (line - i) / i;
138         }
139         printf("\n");
140     }
```

Variables:

- number: Un entero que representa el número de líneas que se mostrarán en el triángulo de Pascal.

Proceso Principal:

- Esta función muestra el Triángulo de Pascal con un número especificado de líneas.
- Solicita al usuario que ingrese el número de líneas que desea en el Triángulo de Pascal.
- Utiliza dos bucles for anidados para imprimir el triángulo.
- El bucle externo (line) controla el número de líneas y se encarga de imprimir los espacios en blanco antes de cada línea.
- El bucle interno (i) calcula y muestra los coeficientes binomiales para cada línea del triángulo utilizando el valor de coef.

}