Reporte de practica 5

González Pardo Adrian

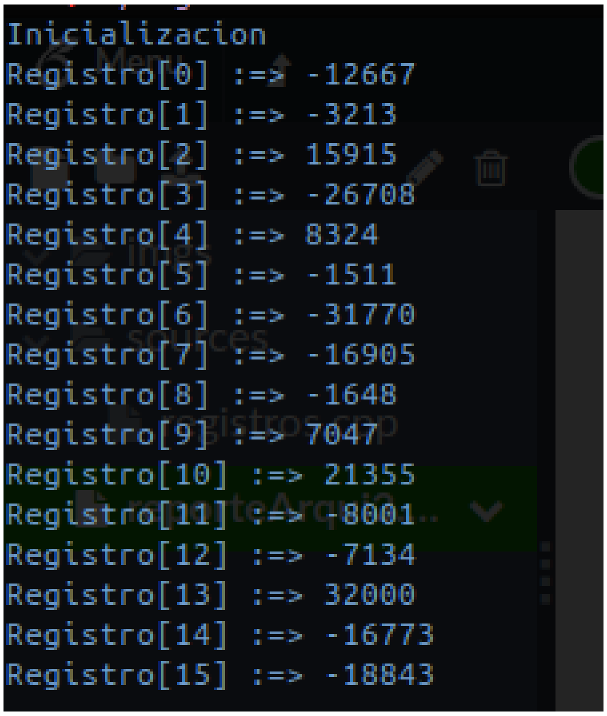
Marzo 2020

1. Código C++

```
/*
 * Alumno: Gonzalez Pardo Adrian
 * Grupo: 3CV8
 * Practica 5
 * Desarrollado para Linux por el define de colores
 */
#include <bits/stdc++.h>
using namespace std;

#define KNRM  "\x1B[0m"
#define KRED  "\x1B[31m"
#define KGRN  "\x1B[32m"
#define KYEL  "\x1B[33m"
#define KBLU  "\x1B[34m"
#define KMAG  "\x1B[35m"
#define KCYN  "\x1B[36m"
#define KWHT  "\x1B[37m"
#define BRED  "\x1B[91m"
#define BGRN  "\x1B[92m"
#define BYEL  "\x1B[93m"
#define BBLU  "\x1B[94m"
#define BMAG  "\x1B[95m"
#define BCYN  "\x1B[96m"
#define BWHT  "\x1B[97m"

class registros{
  private:
    int banco[16],writeData,writeReg,readData1,readData2,shamt;
    bool WR,SHE,DIR,CLR;
  public:
    /* Constructor */
    registros(){
      srand(time(NULL));
    }

    /* Getters and Setters */
    int getWriteReg(){return this->writeReg;}

    int getWriteData(){return this->writeData;}

    int getReadData1(){return this->readData1;}

    int getReadData2(){return this->readData2;}

    int getShamt(){return this->shamt;}

    bool isWR(){return this->WR;}

    bool isSHE(){return this->SHE;}

    bool isDIR(){return this->DIR;}

    bool isCLR(){return this->CLR;}

    void setWriteReg(int writeReg){this->writeReg=writeReg;}

    void setWriteData(int writeData){this->writeData=writeData;}

    void setReadData1(int readData1){this->readData1=readData1;}

    void setReadData2(int readData2){this->readData2=readData2;}

    void setShamt(int shamt){this->shamt=shamt;}

    void setWR(bool WR){this->WR=WR;}

    void setSHE(bool SHE){this->SHE=SHE;}

    void setDIR(bool DIR){this->DIR=DIR;}

    void setCLR(bool CLR){this->CLR=CLR;}

    /* Set banco with random number between -32768 to 32767 */
    void set(){
      for(int i=0; i<16; i++) {
        /* Limite positivo 32767
         * Limite negativo -32768 */
        *(banco+i)=(rand()%65536)-32768;
      }
    }


    /* Validation function */
    bool isInReg(int reg){
      return (reg<0 || reg>16);
    }

    bool isReg(int reg){
      return (reg>0&&reg<16);
    }

    /* Get banco data */
    void get(){
      for(int i=0; i<16; i++){
        cout<< "Registro["<<i<<"] :=> "<<banco[i]<<"\n";
      }
    }


    /* Operaciones del banco de registros */
    void operacionSincrona(int writeData,int writeReg,
        int readReg1,int readReg2,int shamt,
        bool WR,bool SHE, bool DIR,
        bool CLR){
      setWR(WR);
      setSHE(SHE);
      setDIR(DIR);
      setCLR(CLR);
      setWriteReg(writeReg);
      setWriteData(writeData);
      setShamt(shamt);
      if(getWriteReg()<0 || getWriteReg()>16){
        cout<<BRED<<"Fuera del limite de registros\n"<<KNRM;
        return;
      }

      if(getWriteData()<-32768 || getWriteData()>32767){
        cout<<BRED<<"Valor mayor a un Slit16\n"<<KNRM;
        return;
      }

      if(isInReg(readReg1)){
        cout<<BRED<<"Reg1 fuera del limite de registros\n"<<KNRM;
        return;
      }

      if(isInReg(readReg2)){
        cout<<BRED<<"Reg2 fuera del limite de registros\n"<<KNRM;
        return;
      }

      if(isInReg(getShamt())){
        cout<<BRED<<"Shamt fuera del limite de registros\n"<<KNRM;
        return;
      }

      setReadData1(banco[readReg1]);
      setReadData2(banco[readReg2]);
      if(isCLR()){
        operacionAsincrona(isCLR());
        return;
      }else if(isWR() && !isSHE()){
        *(banco+getWriteReg())=getWriteData();
        return;
      }else if(isWR() && isSHE() && !isDIR()){
        *(banco+getWriteReg())=(*(banco+readReg1)>>getShamt()) & 0
  x0000ffff;
        return;
      }else if(isWR() && isSHE() && isDIR()){
        *(banco+getWriteReg())=(*(banco+readReg1)<<getShamt()) & 0
  x0000ffff;
        return;
      }
    }

    /* Operacion que manda a 0 todo el banco de registros */
    void operacionAsincrona(bool CLR){
      for(int i=0; i<16; i++){
        *(banco+i)=0;
      }
      get();
    }

    /* Operacion que muestra Registros */
    void operacionAsincrona(bool CLR, int readReg1, int readReg2){
      setCLR(CLR);
      if(isCLR()){
        operacionAsincrona(isCLR());
      }
      if(isReg(readReg1)){
        cout<<BGRN<<"Registro["<<readReg1<<"] :=> "<<banco[readReg1]<<"\n"
  <<KNRM;
      }
      if(isReg(readReg2)){
        cout<<BYEL<<"Registro["<<readReg2<<"] :=> "<<banco[readReg2]<<"\n"
  <<KNRM;
      }

    }
};

int main(void) {
  registros r;
  cout<<BBLU<<"Inicializacion\n";
  r.set();
  r.get();
  cout<<BCYN<<"\n\t\t(Operacion 1)\n\toperacionAsincrona(1) <==> RESET\n";
  r.operacionAsincrona(1);

  cout<<KGRN<<"\n\t\t(Operacion 2)\n\tBANCO[1]=89 <==> operacionSincrona
    (89,1,0,0,0,1,0,0,0)\n";
  r.operacionSincrona(89,1,0,0,0,1,0,0,0);
  r.get();

  cout<<KCYN<<"\n\t\t(Operacion 3)\n\tBANCO[2]=72 <==> operacionSincrona
    (72,2,0,0,0,1,0,0,0)\n";
  r.operacionSincrona(72,2,0,0,0,1,0,0,0);
  r.get();

  cout<<KGRN<<"\n\t\t(Operacion 4)\n\tBANCO[3]=123 <==> operacionSincrona
    (123,3,0,0,0,1,0,0,0)\n";
  r.operacionSincrona(123,3,0,0,0,1,0,0,0);
  r.get();

  cout<<KYEL<<"\n\t\t(Operacion 5)\n\tBANCO[4]=53 <==> operacionSincrona
    (53,4,0,0,0,1,0,0,0)\n";
  r.operacionSincrona(53,4,0,0,0,1,0,0,0);
  r.get();

  cout<<BMAG<<"\n\t\t(Operacion 6)\n\tREAD BANCO[1] & BANCO[2]\n";
  r.operacionAsincrona(0,1,2);

  cout<<KMAG<<"\n\t\t(Operacion 7)\n\tREAD BANCO[3] & BANCO[4]\n";
  r.operacionAsincrona(0,3,4);

  cout<<BBLU<<"\n\t\t(Operacion 8)\n\tBANCO[2]=BANCO[1]<<3 <==>
    operacionSincrona(0,2,1,0,3,1,1,1,0)\n";
  r.operacionSincrona(0,2,1,0,3,1,1,1,0);
  r.get();

  cout<<KYEL<<"\n\t\t(Operacion 9)\n\tBANCO[4]=BANCO[3]>>5 <==>
    operacionSincrona(0,4,3,0,5,1,1,0,0)\n";
  r.operacionSincrona(0,4,3,0,5,1,1,0,0);
  r.get();

  cout<<KGRN<<"\n\t\t(Operacion 10)\n\tREAD BANCO[1] & BANCO[2]\n";
  r.operacionAsincrona(0,1,2);

  cout<<KYEL<<"\n\t\t(Operacion 11)\n\tREAD BANCO[3] & BANCO[4]\n";
  r.operacionAsincrona(0,3,4);

  cout<<KBLU<<"\n\t\t(Operacion 12)\n\tget()\n";
  r.get();

  cout<<BCYN<<"\n\t\t(Operacion 13)\n\toperacionAsincrona(1) <==> RESET\n"
    ;
  r.operacionAsincrona(1);

  cout<<KNRM<<endl;
  return 0;
}
```

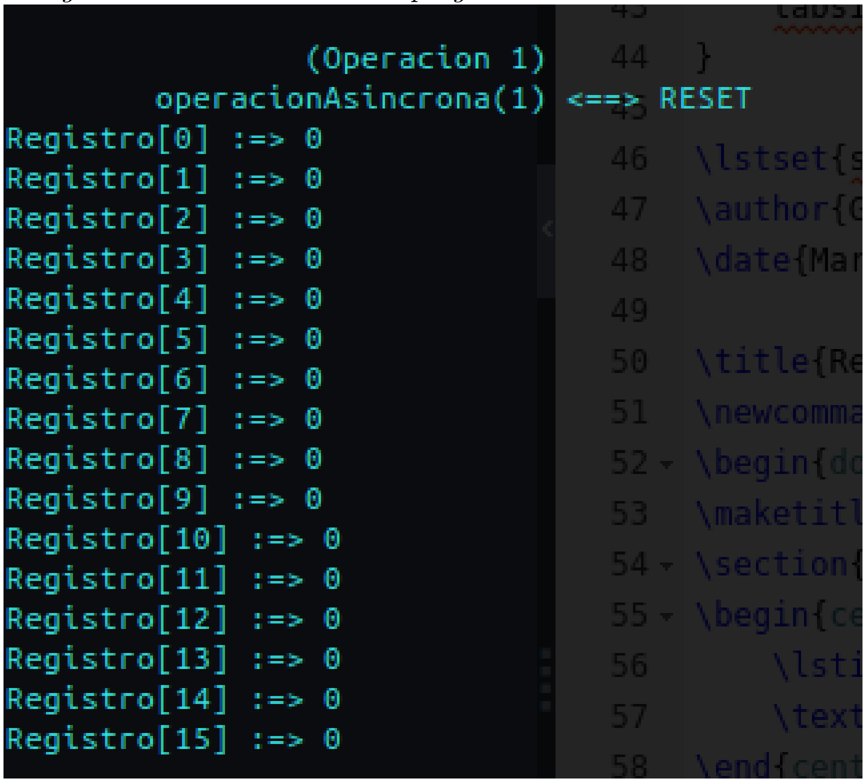
Código fuente de archivo de registros

2. Captura de simulaciones



```
Inicializacion
Registro[0] :=> -12667
Registro[1] :=> -3213
Registro[2] :=> 15915
Registro[3] :=> -26708
Registro[4] :=> 8324
Registro[5] :=> -1511
Registro[6] :=> -31770
Registro[7] :=> -16905
Registro[8] :=> -1648
Registro[9] :=> 7047
Registro[10] :=> 21355
Registro[11] :=> -8001
Registro[12] :=> -7134
Registro[13] :=> 32000
Registro[14] :=> -16773
Registro[15] :=> -18843
```

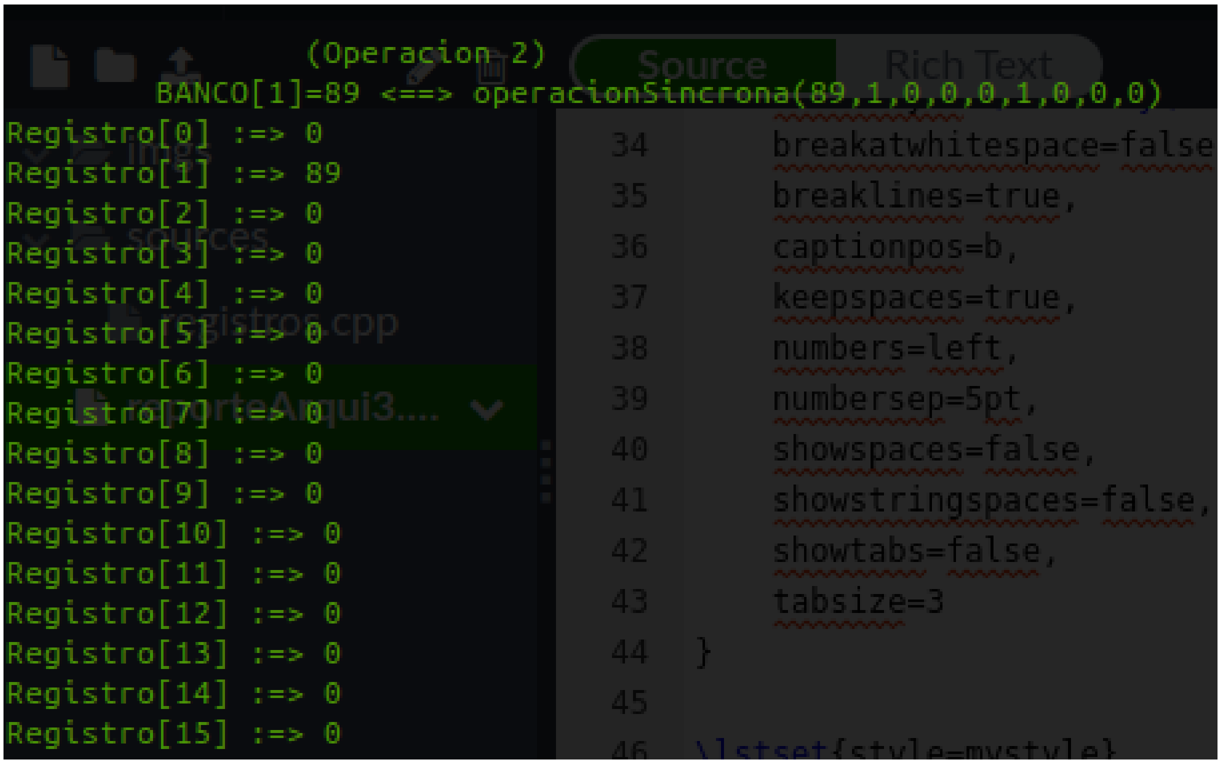
Figura 0: Inicialización del programa con números random



```
                (Operacion 1)
        operacionAsincrona(1) <==> RESET
Registro[0] :=> 0
Registro[1] :=> 0
Registro[2] :=> 0
Registro[3] :=> 0
Registro[4] :=> 0
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

```
}

\lstset{s
\author{G
\date{Mar

\title{Re
\newcomma
\begin{do
\maketitl
\section{
\begin{ce
    \lsti
    \text
\end{cent
```

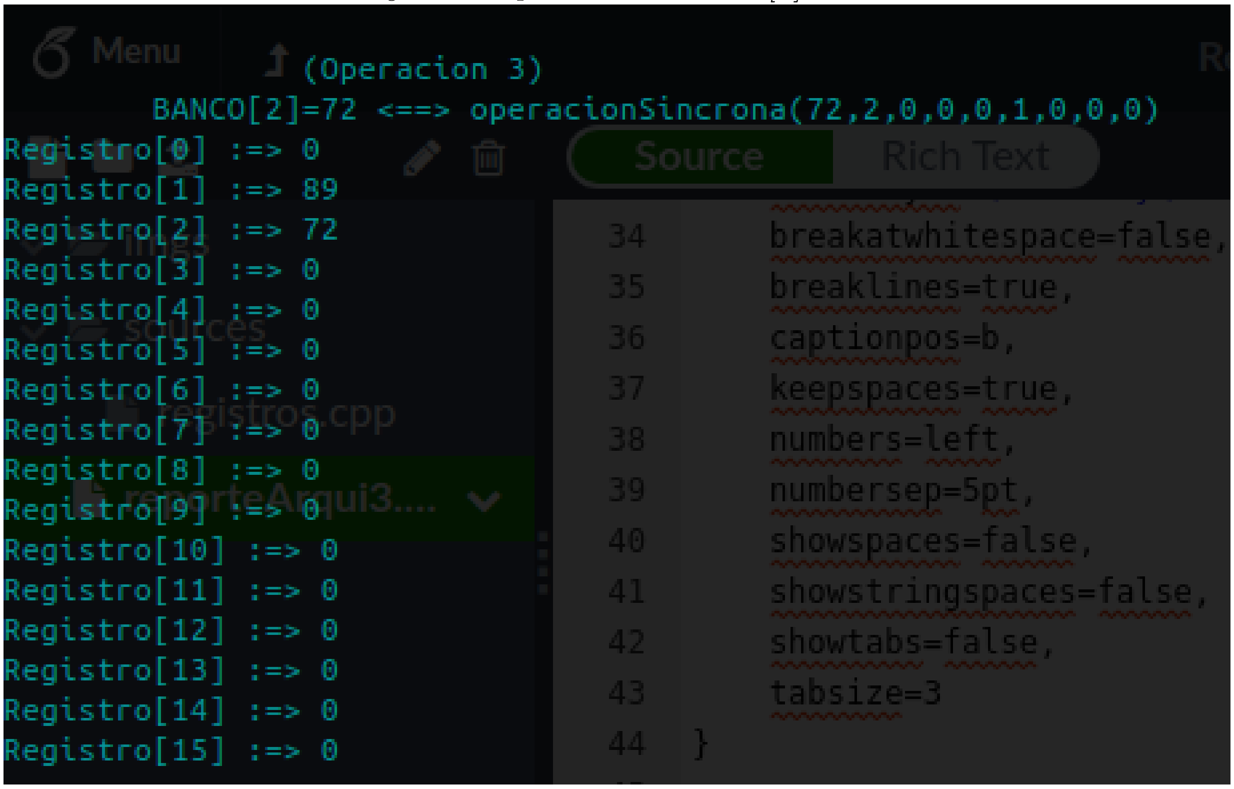
Figura 1: Operación 1 Reset”



```
          (Operacion 2)
  BANCO[1]=89 <===> operacionSincrona(89,1,0,0,0,1,0,0,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 0
Registro[3] :=> 0
Registro[4] :=> 0
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

```
    breakatwhitespace=false
    breaklines=true,
    captionpos=b,
    keepspaces=true,
    numbers=left,
    numbersep=5pt,
    showspaces=false,
    showstringspaces=false,
    showtabs=false,
    tabsize=3
}

\lstset{style=mystyle}
```

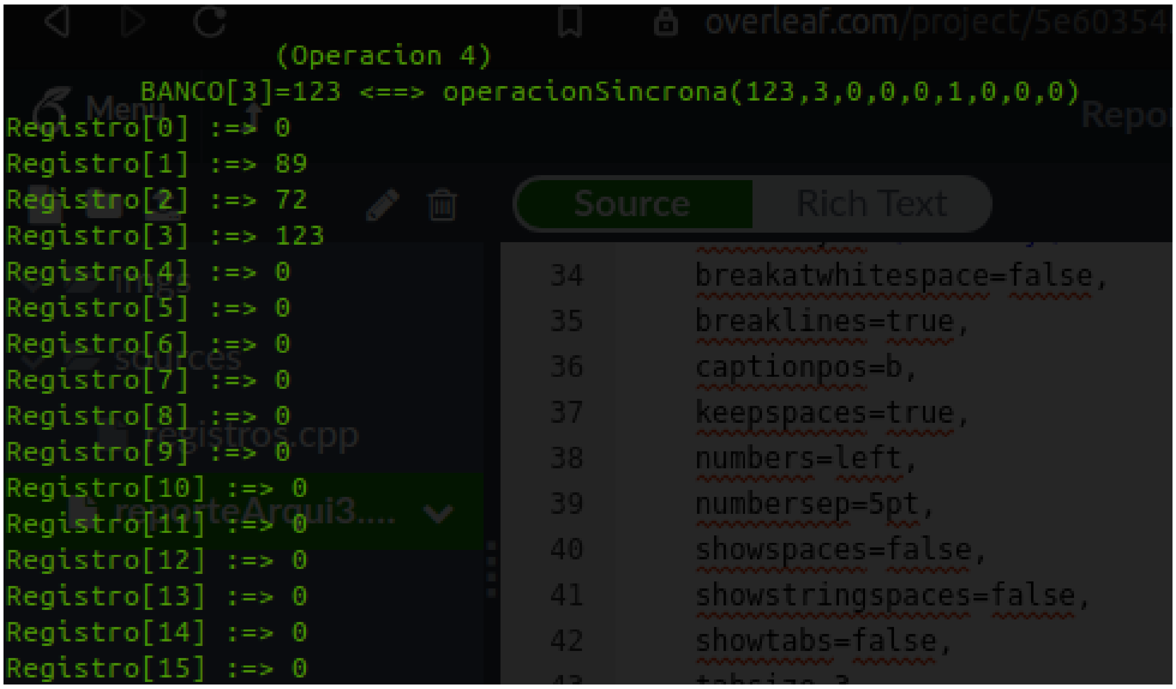
Figura 2: Operación 2 "Banco[1]=89"



```
          (Operacion 3)
  BANCO[2]=72 <===> operacionSincrona(72,2,0,0,0,1,0,0,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 72
Registro[3] :=> 0
Registro[4] :=> 0
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

```
    breakatwhitespace=false,
    breaklines=true,
    captionpos=b,
    keepspaces=true,
    numbers=left,
    numbersep=5pt,
    showspaces=false,
    showstringspaces=false,
    showtabs=false,
    tabsize=3
}
```

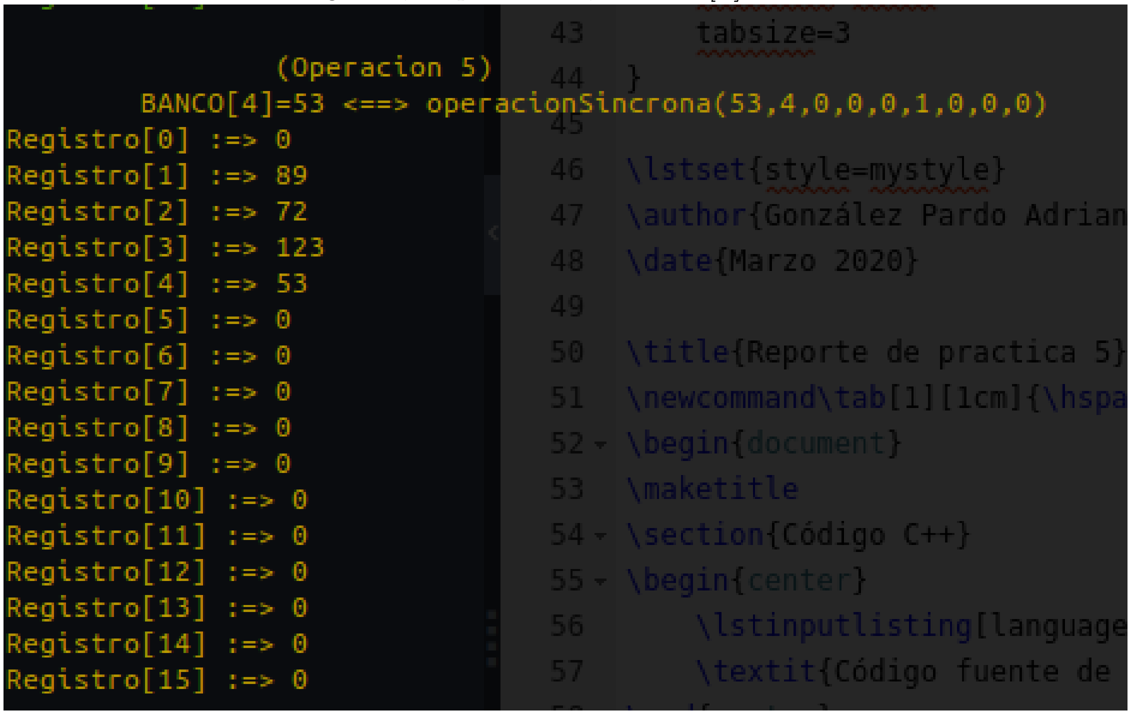
Figura 3: Operación 3 "Banco[2]=72"



```
                (Operacion 4)
        BANCO[3]=123 <==> operacionSincrona(123,3,0,0,0,1,0,0,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 72
Registro[3] :=> 123
Registro[4] :=> 0
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

```
    breakatwhitespace=false,
    breaklines=true,
    captionpos=b,
    keepspaces=true,
    numbers=left,
    numbersep=5pt,
    showspaces=false,
    showstringspaces=false,
    showtabs=false,
```

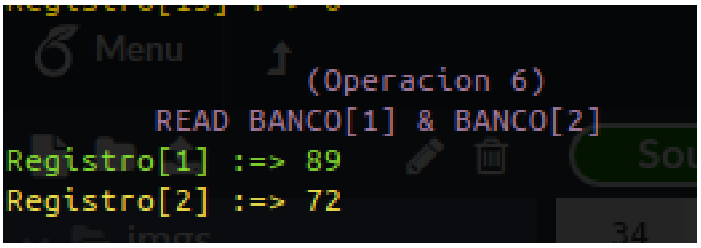
Figura 4: Operación 4 "Banco[3]=123"



```
                (Operacion 5)
        BANCO[4]=53 <==> operacionSincrona(53,4,0,0,0,1,0,0,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 72
Registro[3] :=> 123
Registro[4] :=> 53
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

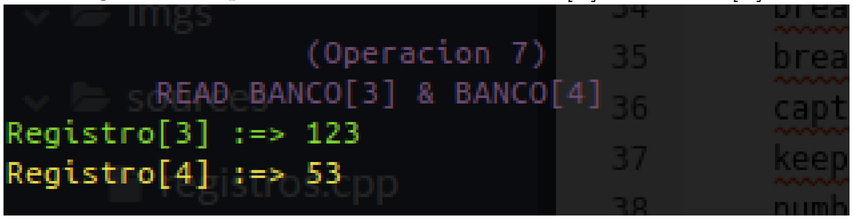
```
    tabsize=3
}

\lstset{style=mystyle}
\author{González Pardo Adrian
\date{Marzo 2020}

\title{Reporte de practica 5}
\newcommand\tab[1][1cm]{\hspa
\begin{document}
\maketitle
\section{Código C++}
\begin{center}
    \lstinputlisting[language
    \textit{Código fuente de
```

Figura 5: Operación 5 "Banco[4]=53"



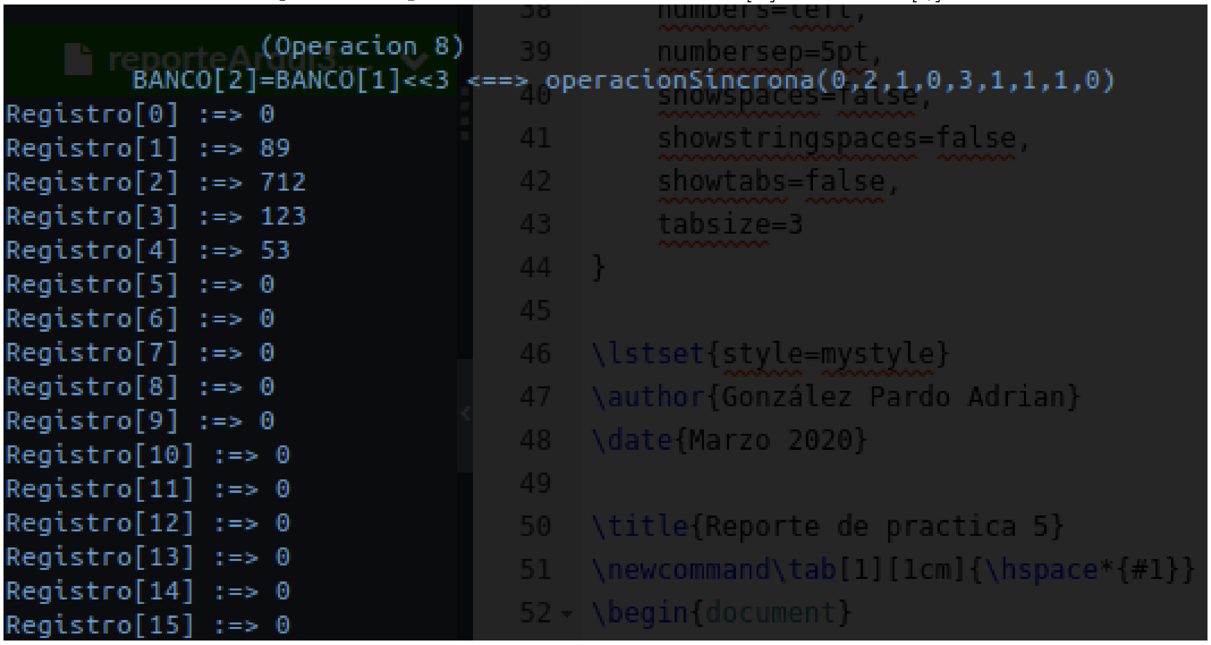
```
        (Operacion 6)
  READ BANCO[1] & BANCO[2]
Registro[1] :=> 89
Registro[2] :=> 72
```

Figura 6: Operación 6 READ Banco[1] & Banco[2]"



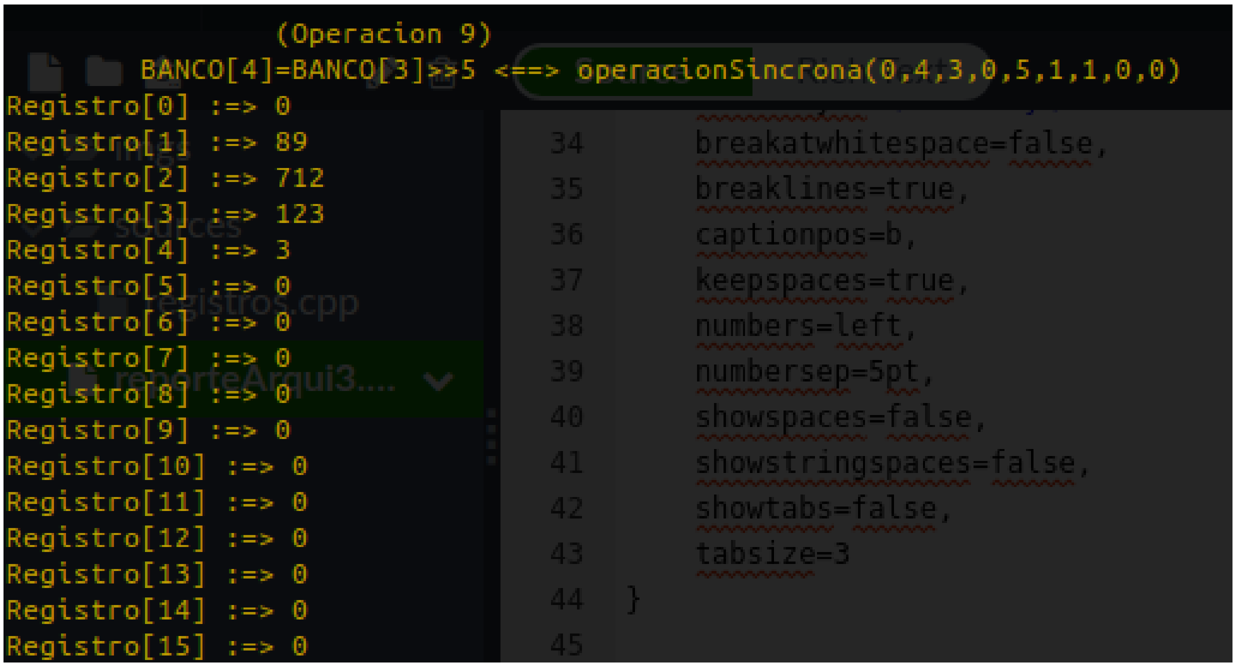
```
            (Operacion 7)
    READ BANCO[3] & BANCO[4]
Registro[3] :=> 123
Registro[4] :=> 53
```

Figura 7: Operación 7 READ Banco[3] & Banco[4]"



```
                (Operacion 8)
        BANCO[2]=BANCO[1]<<3 <==> operacionSincrona(0,2,1,0,3,1,1,1,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 712
Registro[3] :=> 123
Registro[4] :=> 53
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

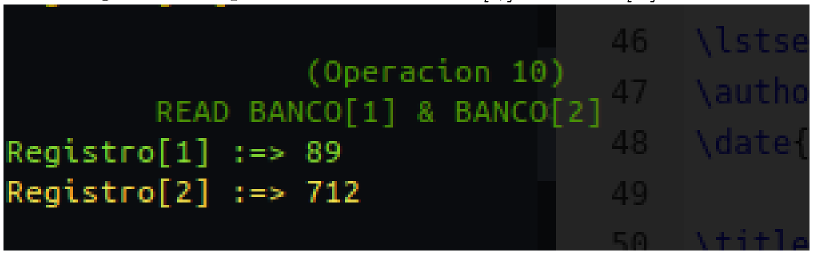
Figura 8: Operación 8 "Banco[2]=Banco[1]<<3"



```
            (Operacion 9)
    BANCO[4]=BANCO[3]>>5 <==> operacionSincrona(0,4,3,0,5,1,1,0,0)
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 712
Registro[3] :=> 123
Registro[4] :=> 3
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

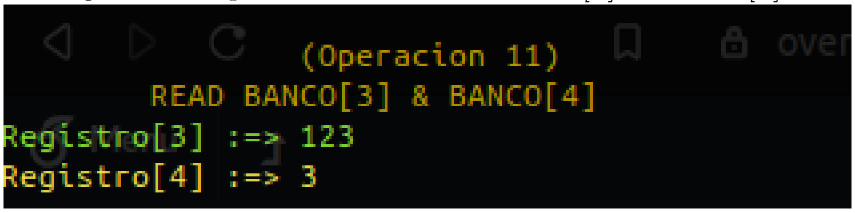
```
    breakatwhitespace=false,
    breaklines=true,
    captionpos=b,
    keepspaces=true,
    numbers=left,
    numbersep=5pt,
    showspaces=false,
    showstringspaces=false,
    showtabs=false,
    tabsize=3
}
```

Figura 9: Operación 9 "Banco[4]=Banco[3]>>5"



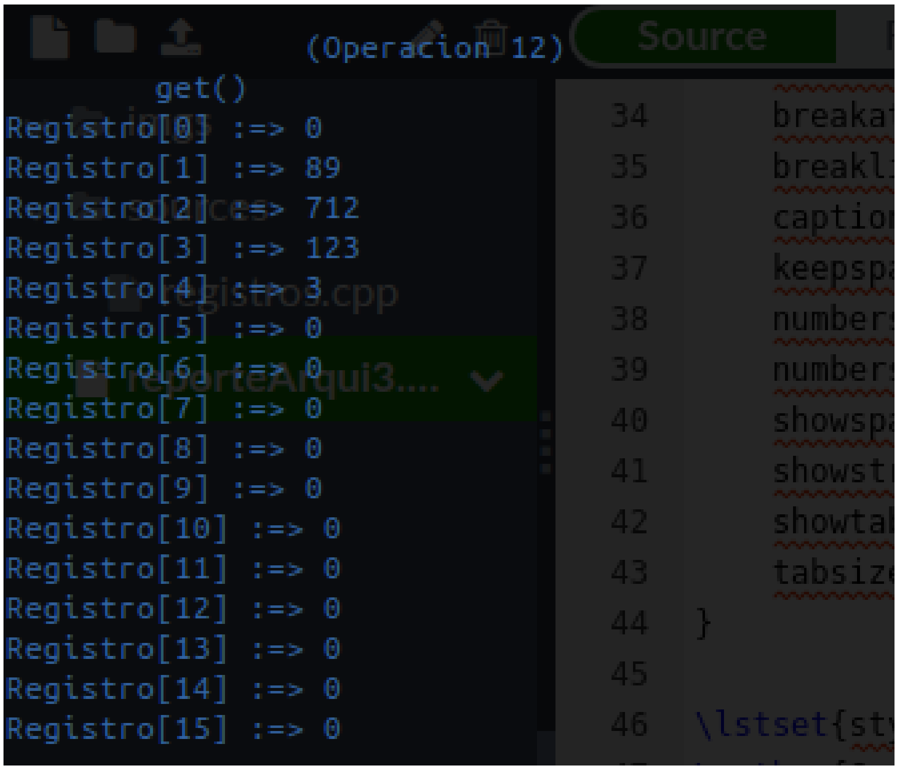
```
              (Operacion 10)
         READ BANCO[1] & BANCO[2]
Registro[1] :=> 89
Registro[2] :=> 712
```

Figura 10: Operación 10 READ Banco[1] & Banco[2]"



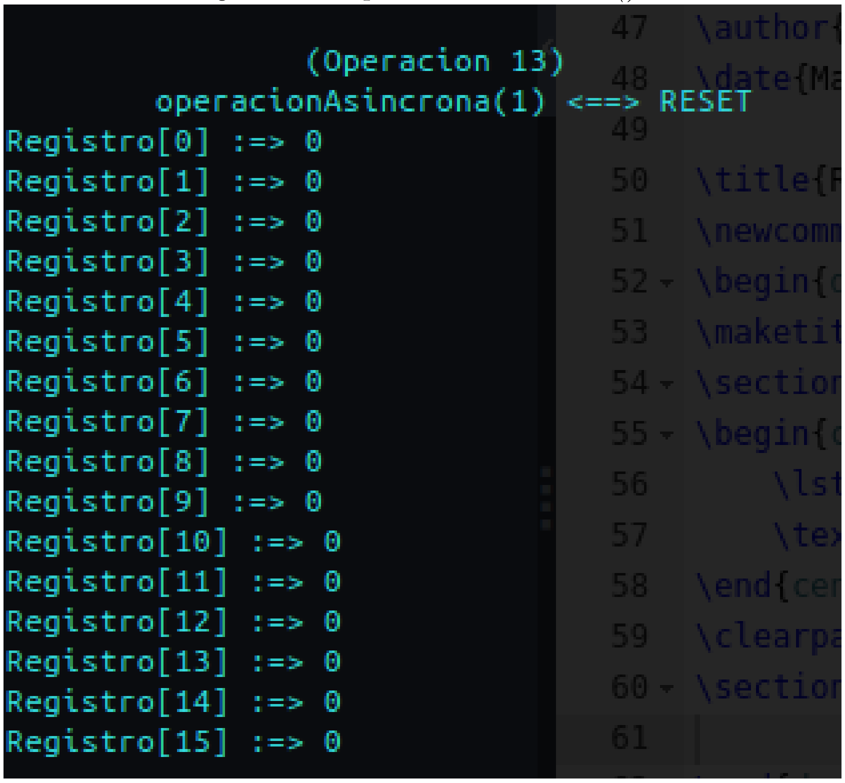
```
              (Operacion 11)
         READ BANCO[3] & BANCO[4]
Registro[3] :=> 123
Registro[4] :=> 3
```

Figura 11: Operación 11 READ Banco[3] & Banco[4]"



```
                    (Operacion 12)
        get()
Registro[0] :=> 0
Registro[1] :=> 89
Registro[2] :=> 712
Registro[3] :=> 123
Registro[4] :=> 3
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

Figura 12: Operación 12 "GET()"



```
                (Operacion 13)
        operacionAsincrona(1) <==> RESET
Registro[0] :=> 0
Registro[1] :=> 0
Registro[2] :=> 0
Registro[3] :=> 0
Registro[4] :=> 0
Registro[5] :=> 0
Registro[6] :=> 0
Registro[7] :=> 0
Registro[8] :=> 0
Registro[9] :=> 0
Registro[10] :=> 0
Registro[11] :=> 0
Registro[12] :=> 0
Registro[13] :=> 0
Registro[14] :=> 0
Registro[15] :=> 0
```

Figura 13: Operación 13 Reset"