INFORMATICA I

Ordenamiento de un vector por el metodo burbuja

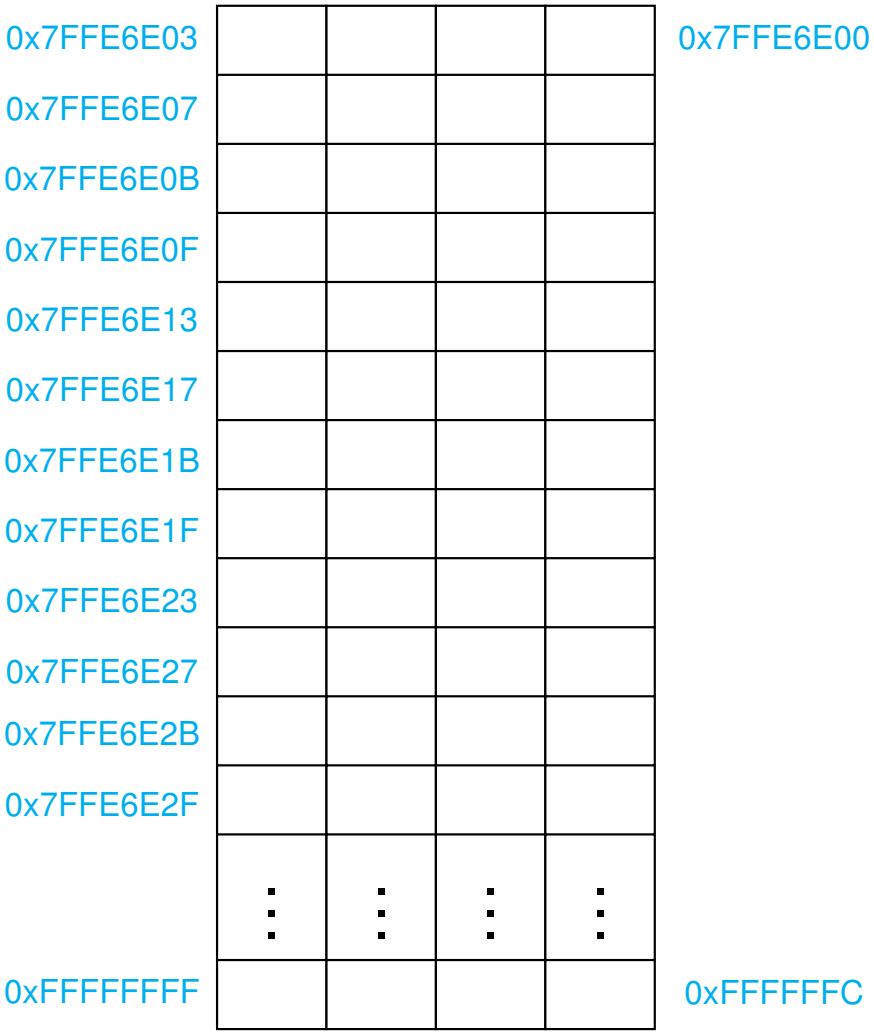
Profesor: *Ing.Jerónimo Atencio*
Auxiliares: *Ing.Alexander Jiricny*
Ing.Juan Carlos Cuttitta

Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Departamento de Ingeniería Electrónica

18 de mayo de 2016

Ordenamiento de un vector por el metodo ***burbuja***

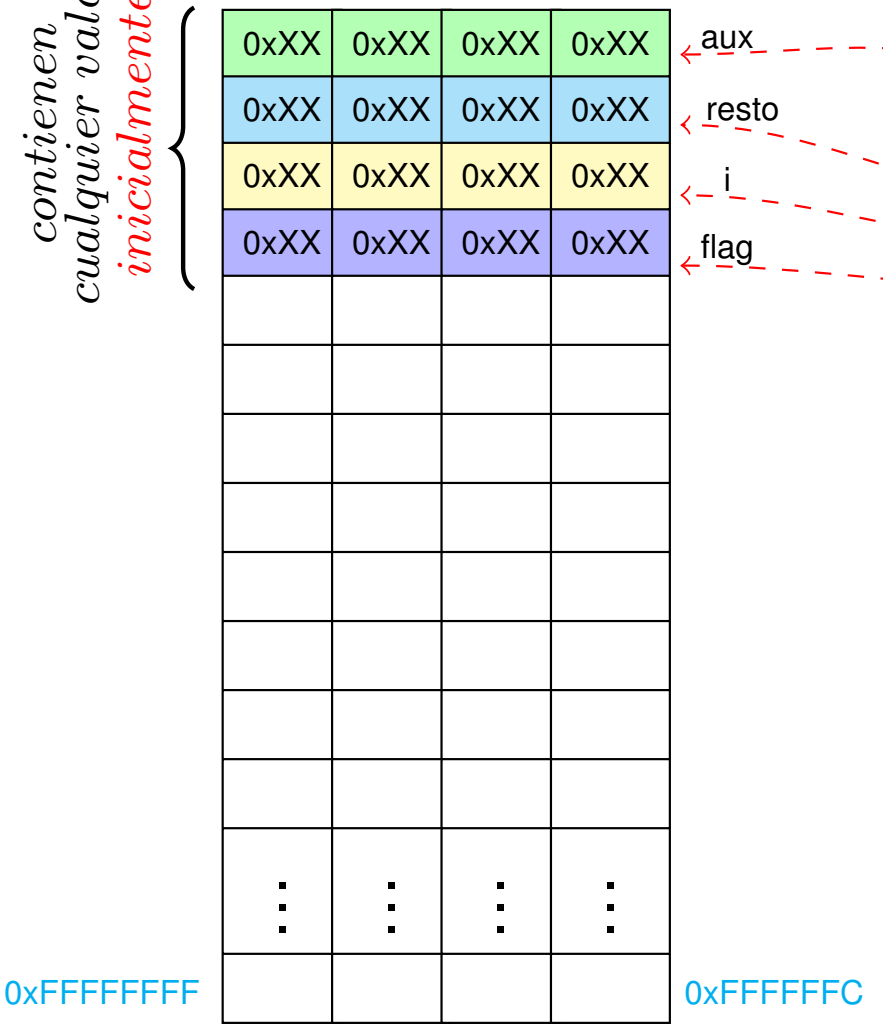
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

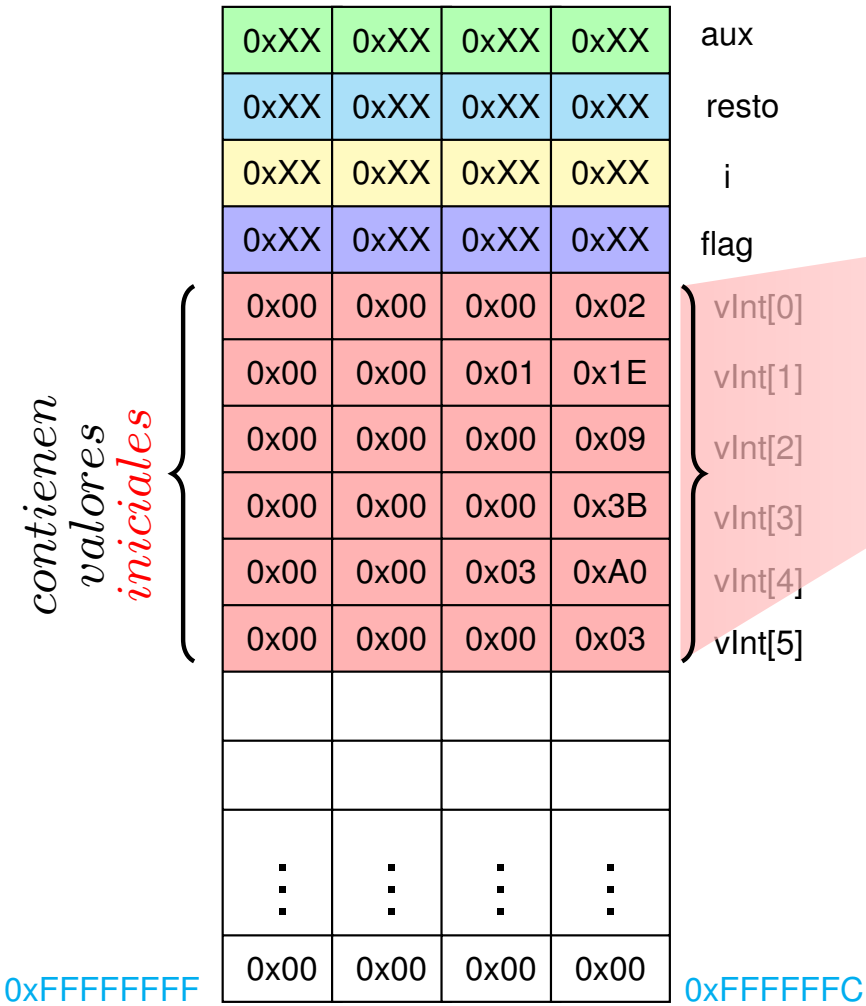
MEMORIA en ARQUITECTURA x86 32-bits

contienen cualquier valor inicialmente



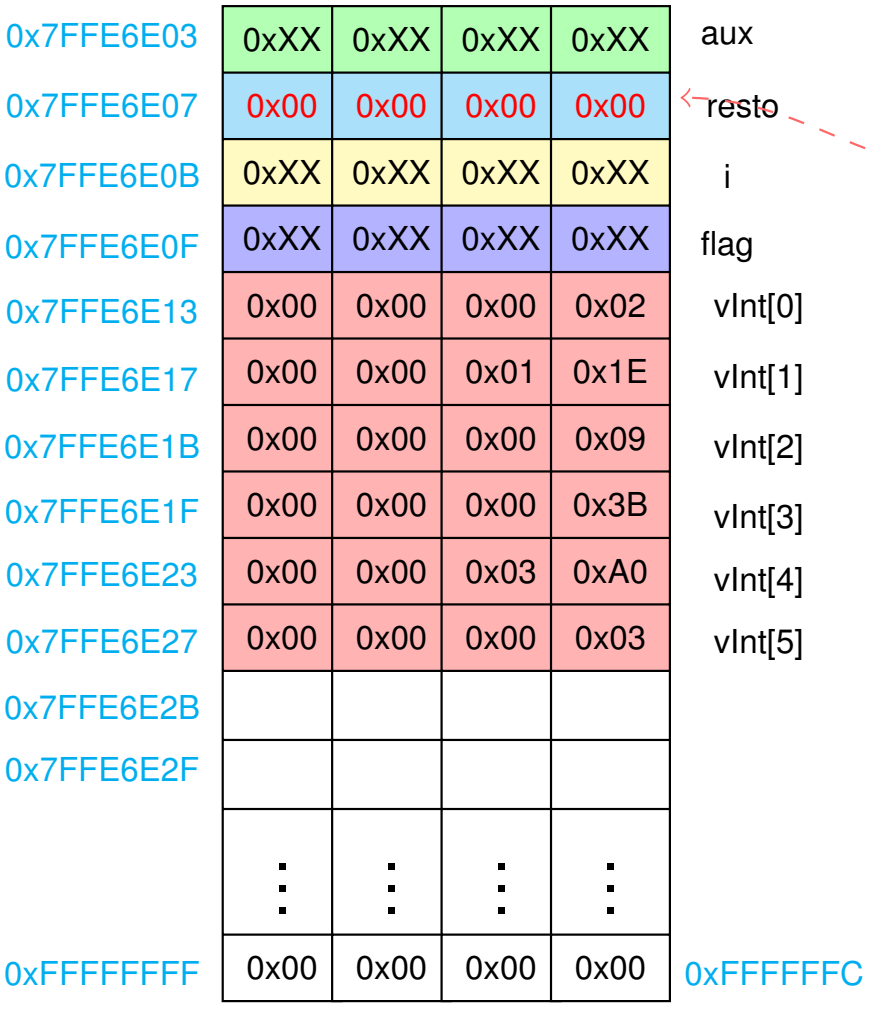

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits



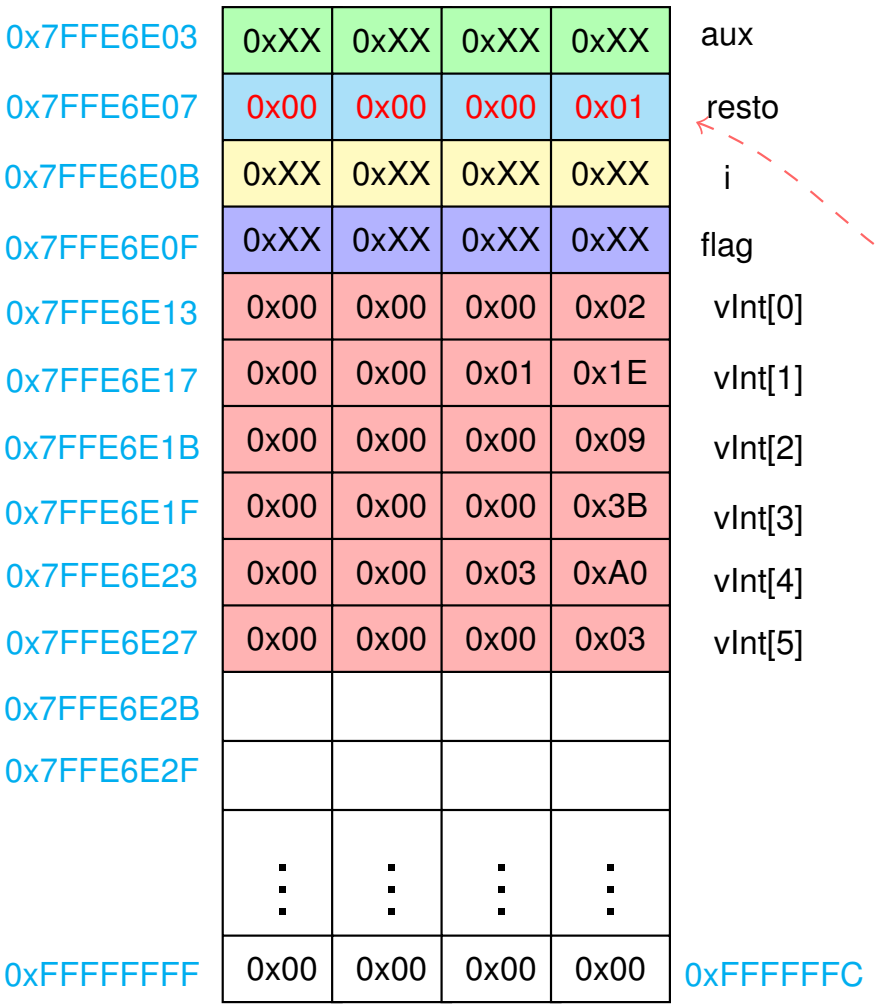

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits



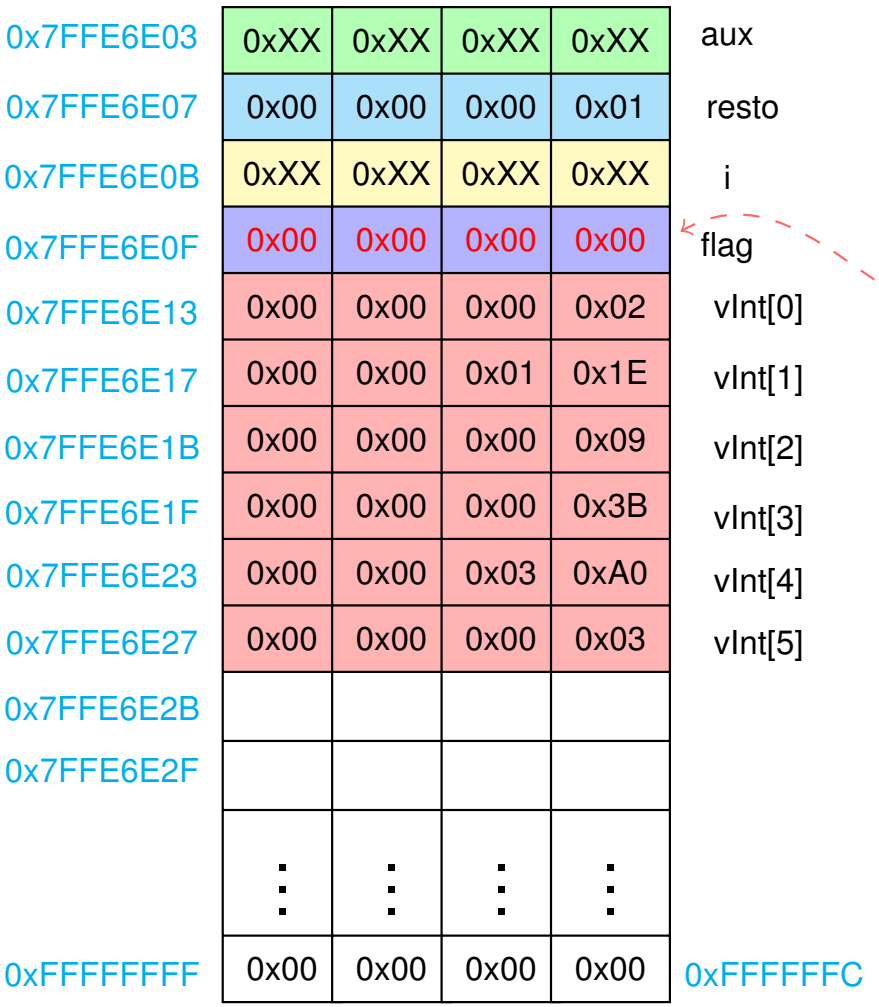

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits



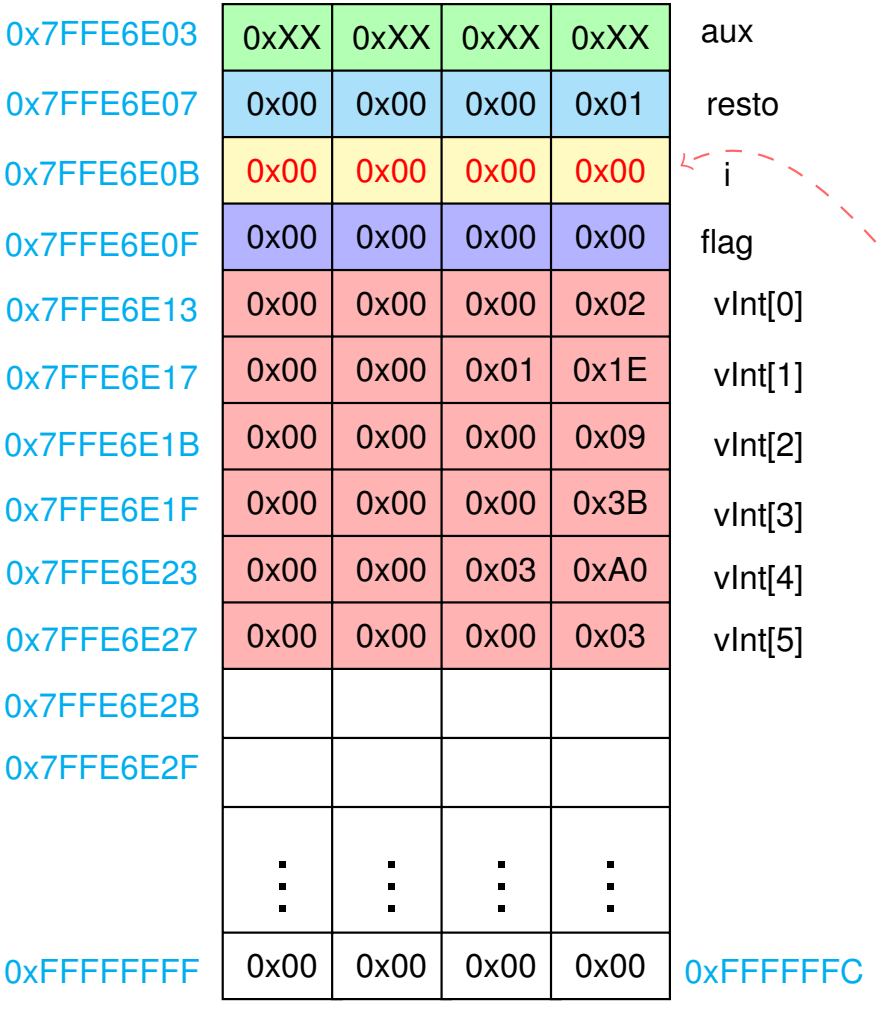

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

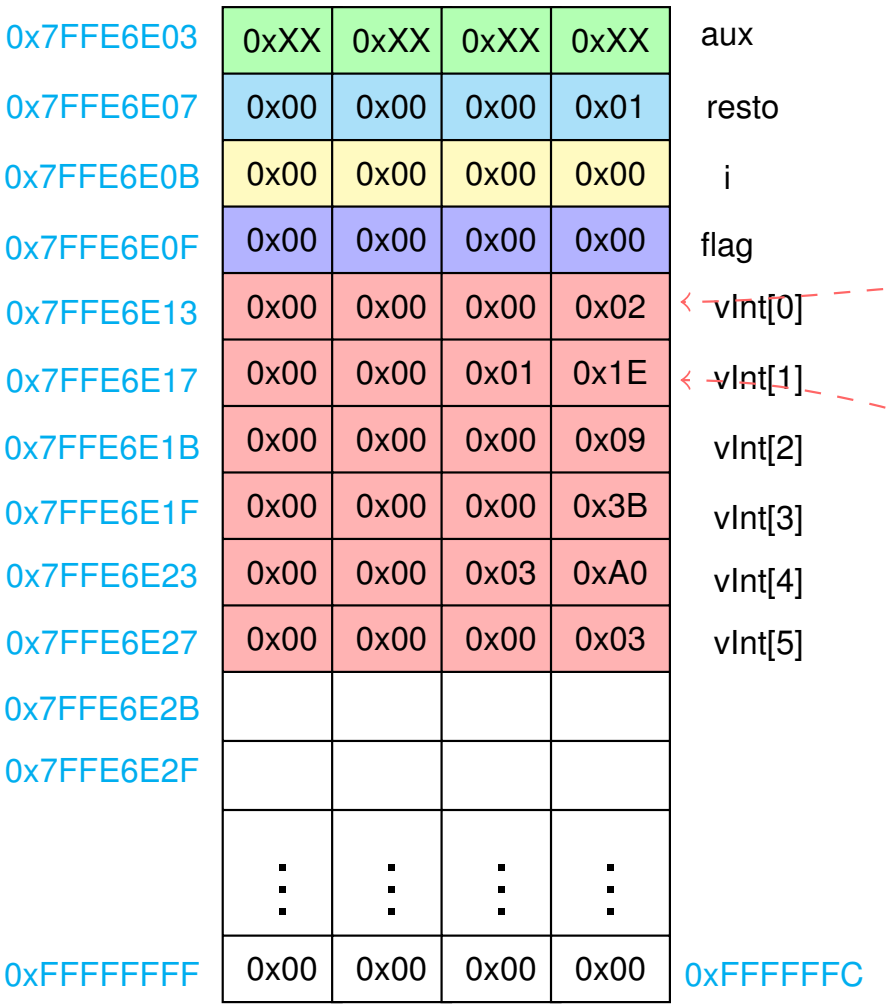
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

MEMORIA en ARQUITECTURA x86 32-bits



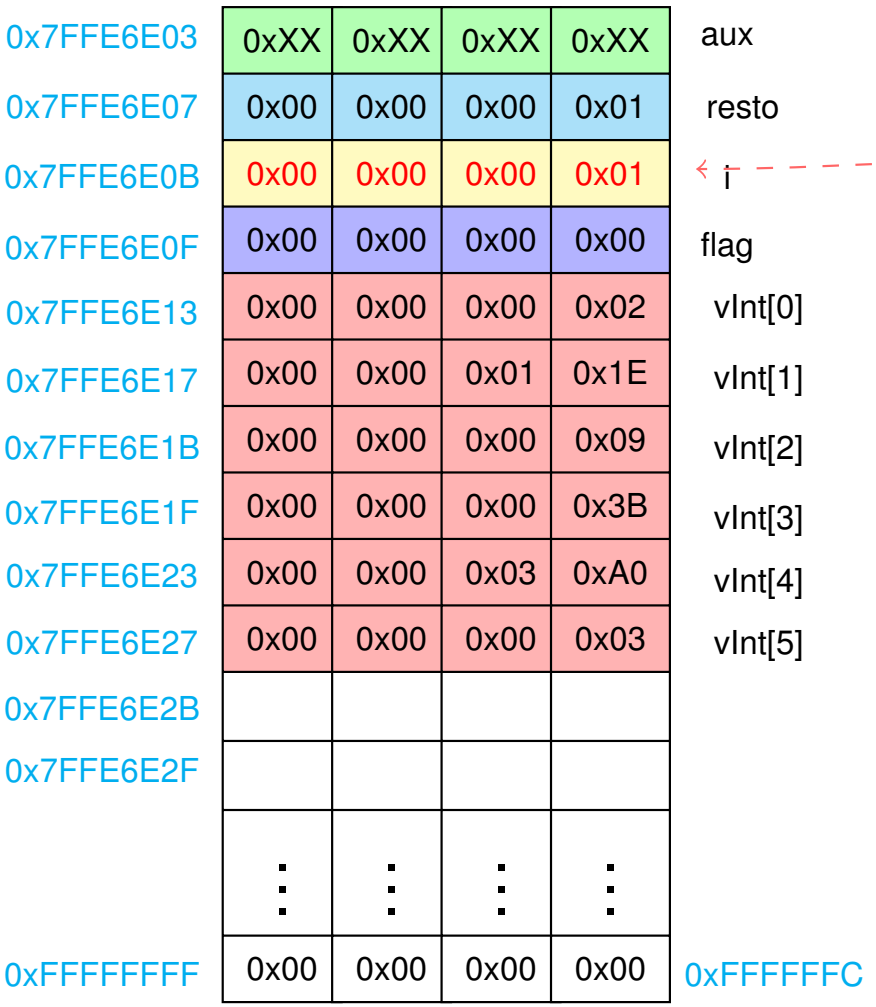

```c
#include <stdio.h>
#define CANT 6

int main (void)
{
int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
  resto=0;
  do
  {
    resto++;
    flag=0;
    for(i=0;i < (CANT-resto);i++)
    {
      if(vInt[i] > vInt[i+1])
      {
        aux=vInt[i];
        vInt[i]=vInt[i+1];
        vInt[i+1]=aux;
        flag=1;
      }
    }
  }while(flag);
return (0);
}
```

6-1=5

0 1

MEMORIA en ARQUITECTURA x86 32-bits

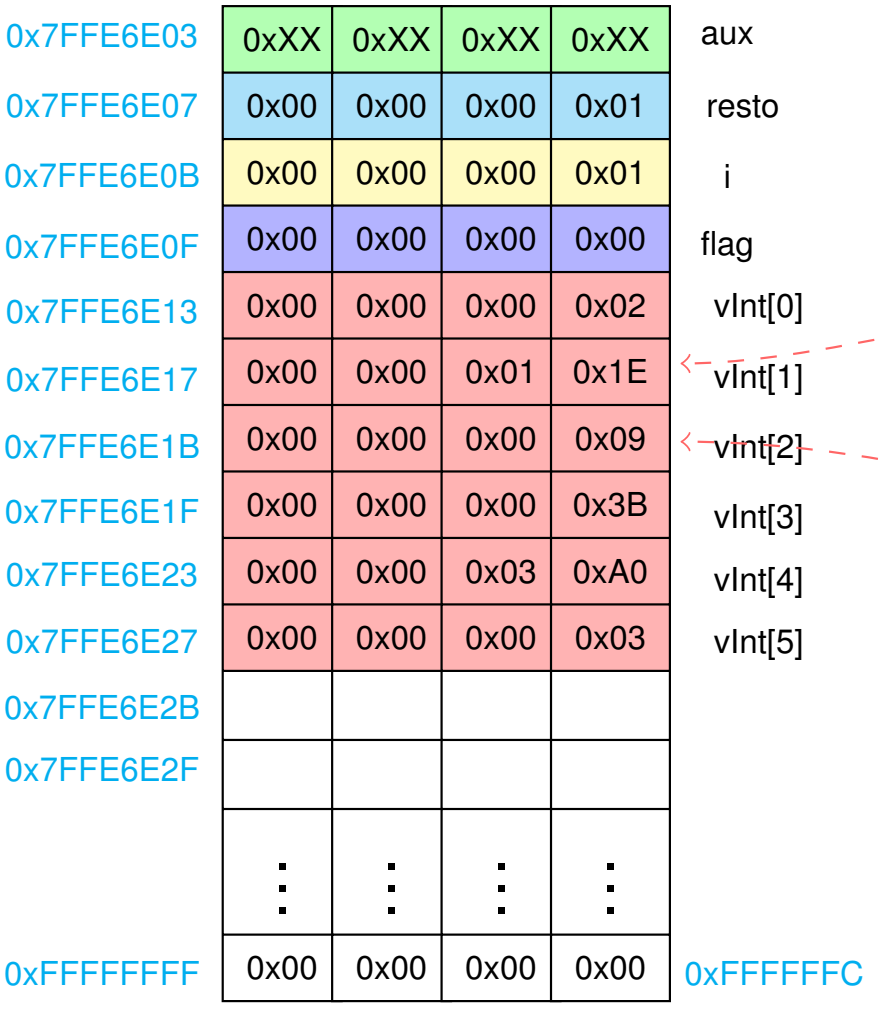



```c
#include <stdio.h>
#define CANT 6

int main (void)
{
int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
  resto=0;
  do
  {
    resto++;
    flag=0;
    for(i=0;i < (CANT-resto);i++)
    {
      if(vInt[i] > vInt[i+1])
      {
        aux=vInt[i];
        vInt[i]=vInt[i+1];
        vInt[i+1]=aux;
        flag=1;
      }
    }
  }while(flag);
return (0);
}
```

6-1=5

MEMORIA en ARQUITECTURA x86 32-bits



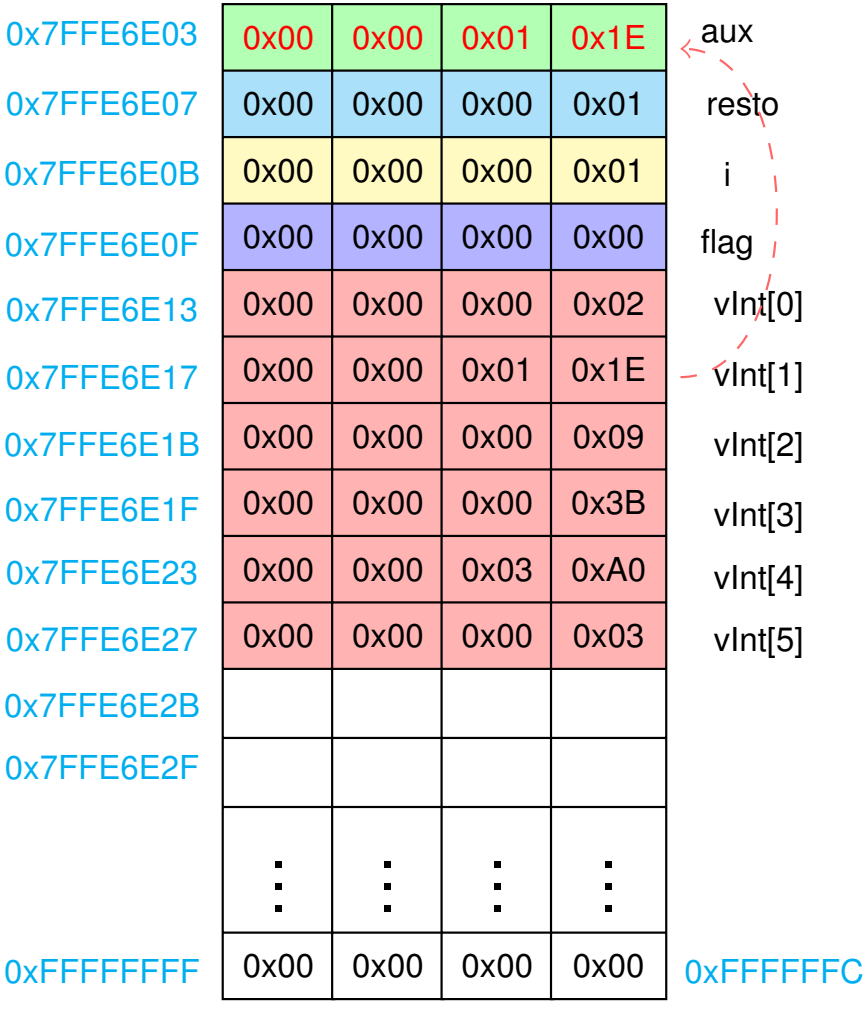

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

1

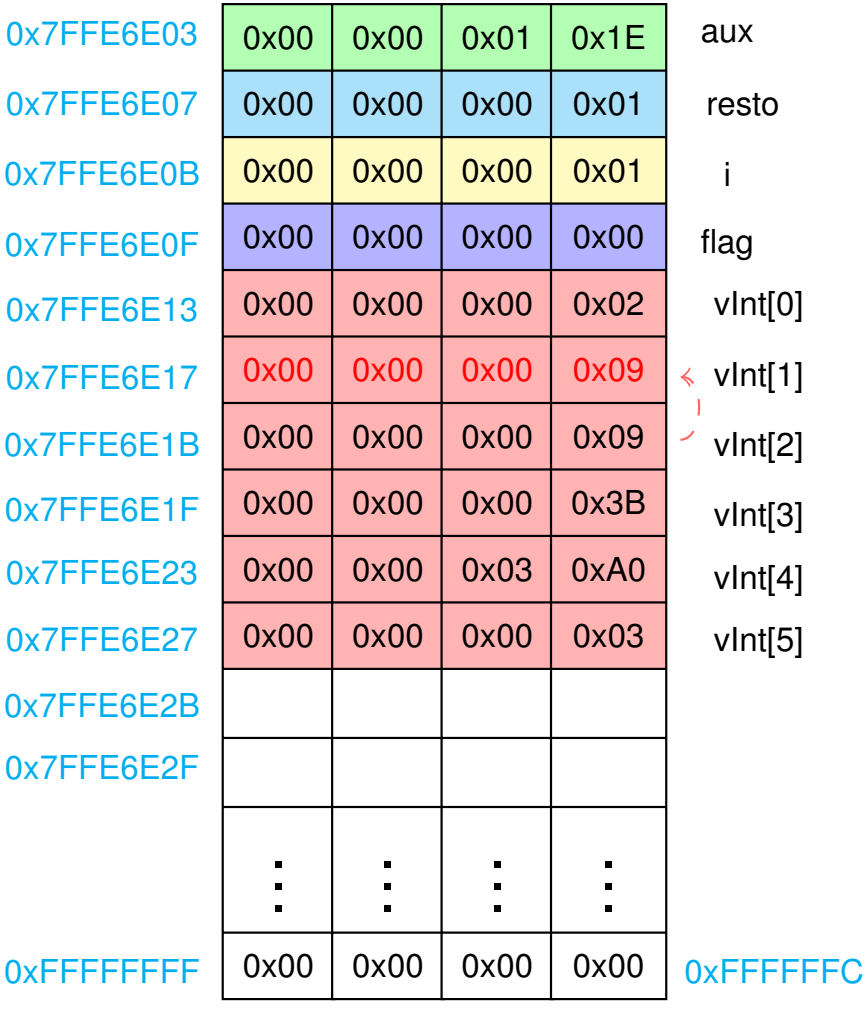
2

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {              1
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits

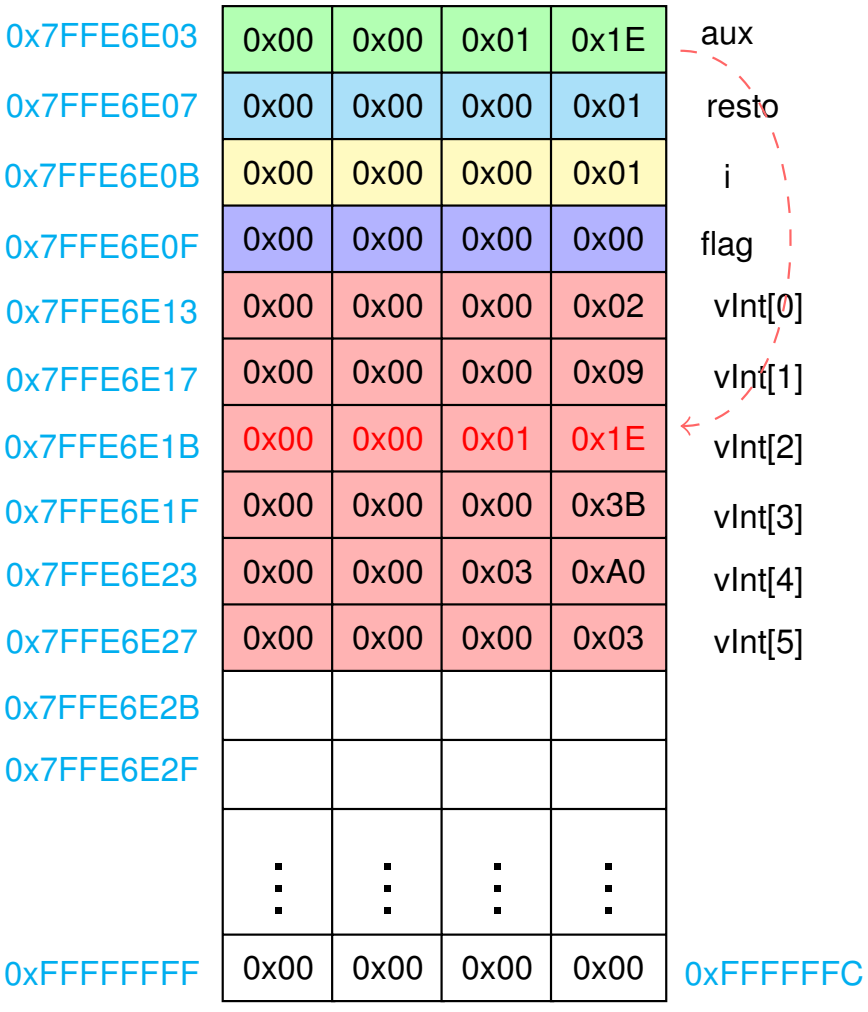



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

vInt[1] = vInt[2]

MEMORIA en ARQUITECTURA x86 32-bits

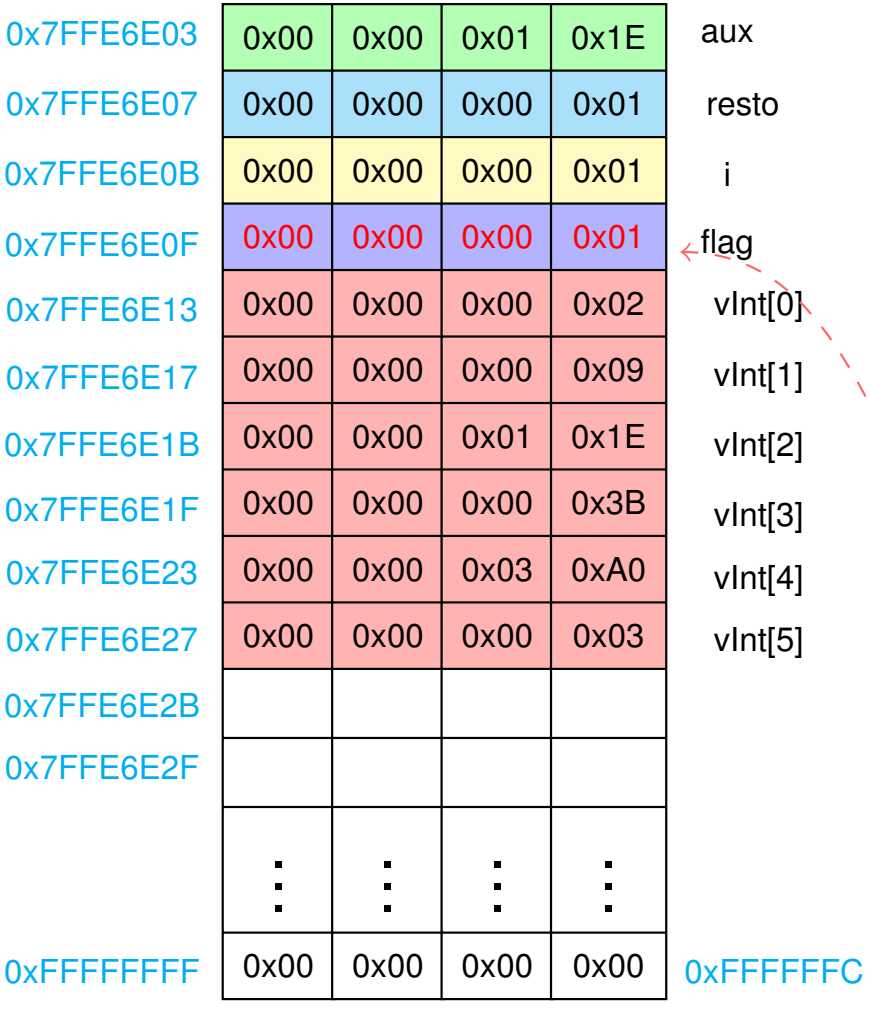



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

vInt[2] = aux

MEMORIA en ARQUITECTURA x86 32-bits

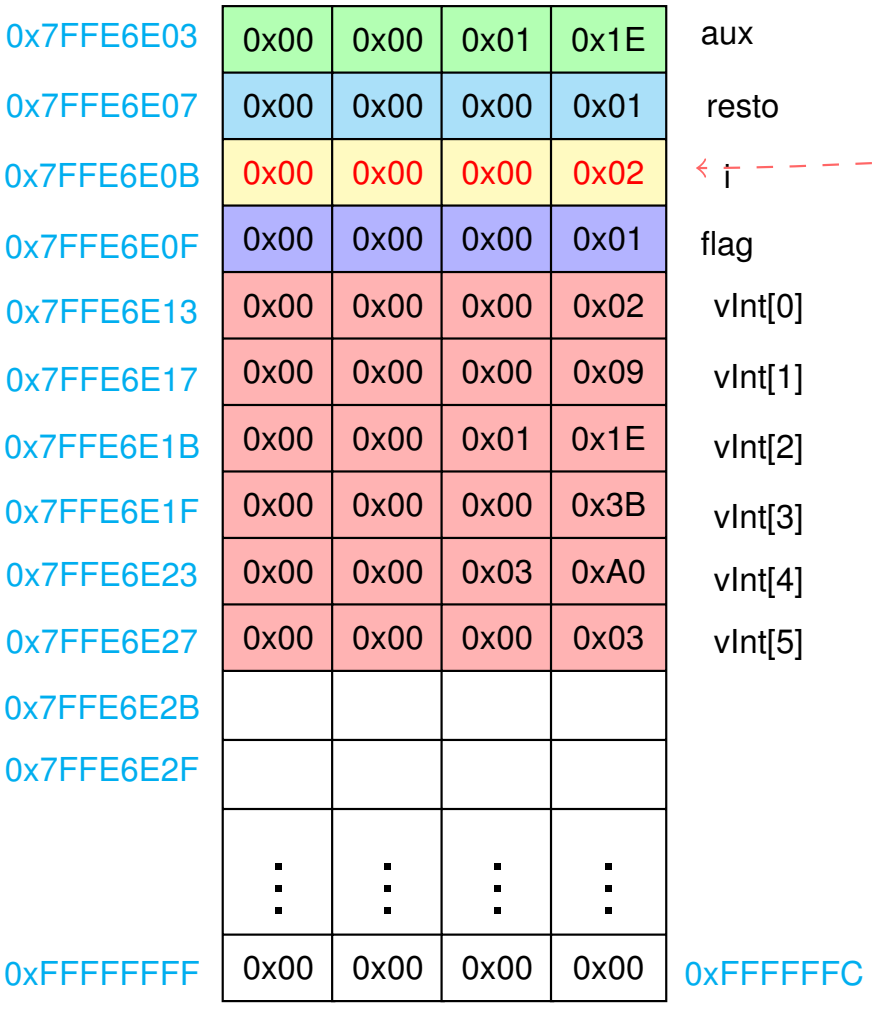



```c
#include <stdio.h>
#define CANT 6

int main (void)
{
int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
  resto=0;
  do
  {
    resto++;
    flag=0;
    for(i=0;i < (CANT-resto);i++)
    {
      if(vInt[i] > vInt[i+1])
      {
        aux=vInt[i];
        vInt[i]=vInt[i+1];
        vInt[i+1]=aux;
        flag=1;
      }
    }
  }while(flag);
return (0);
}
```

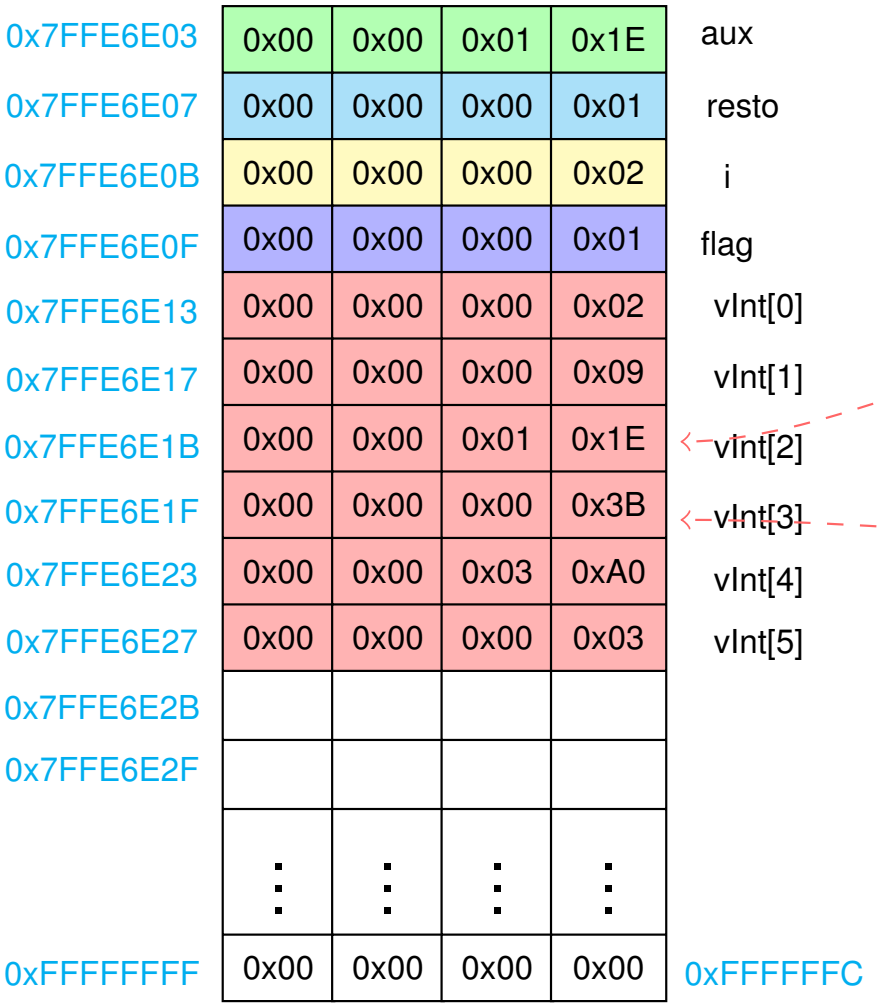
6-1=5

MEMORIA en ARQUITECTURA x86 32-bits



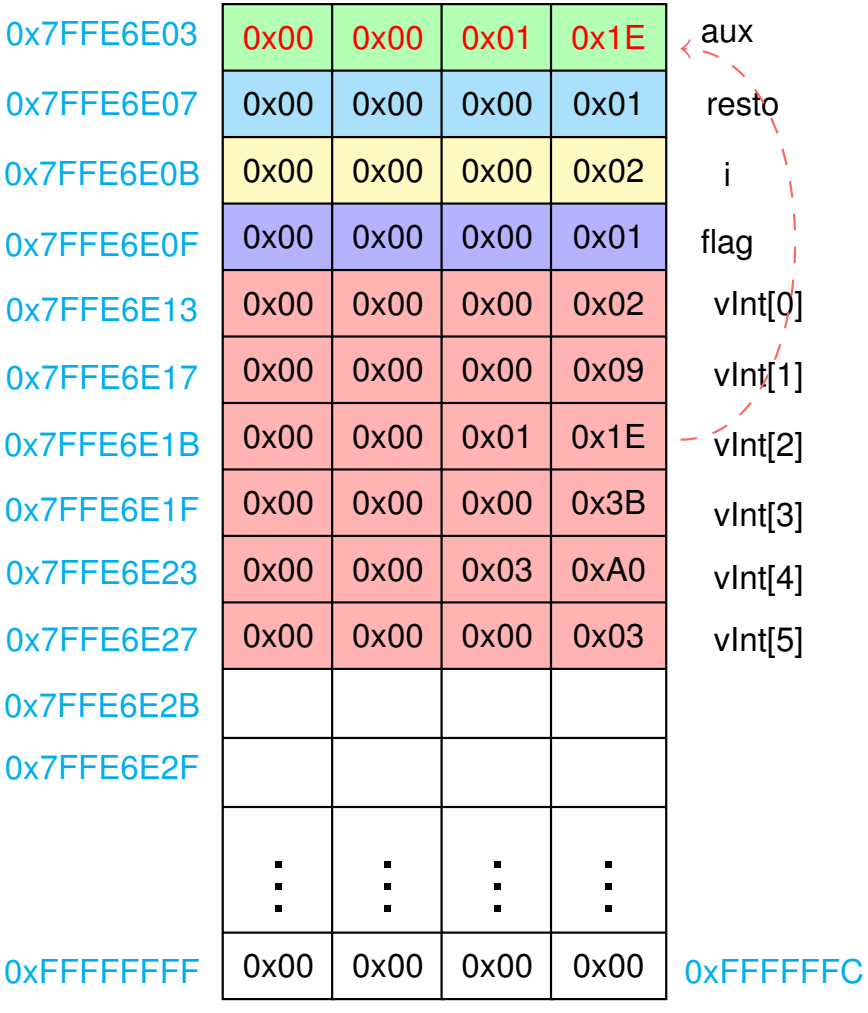

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits



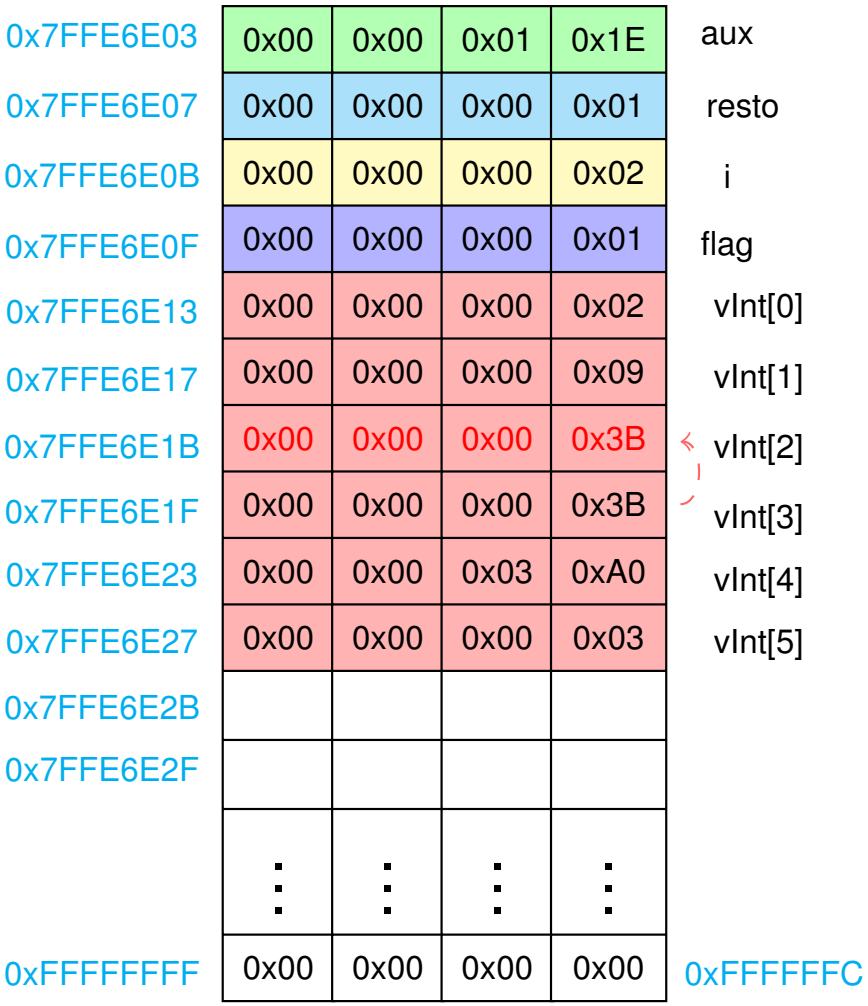

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {          2                3
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {          2
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

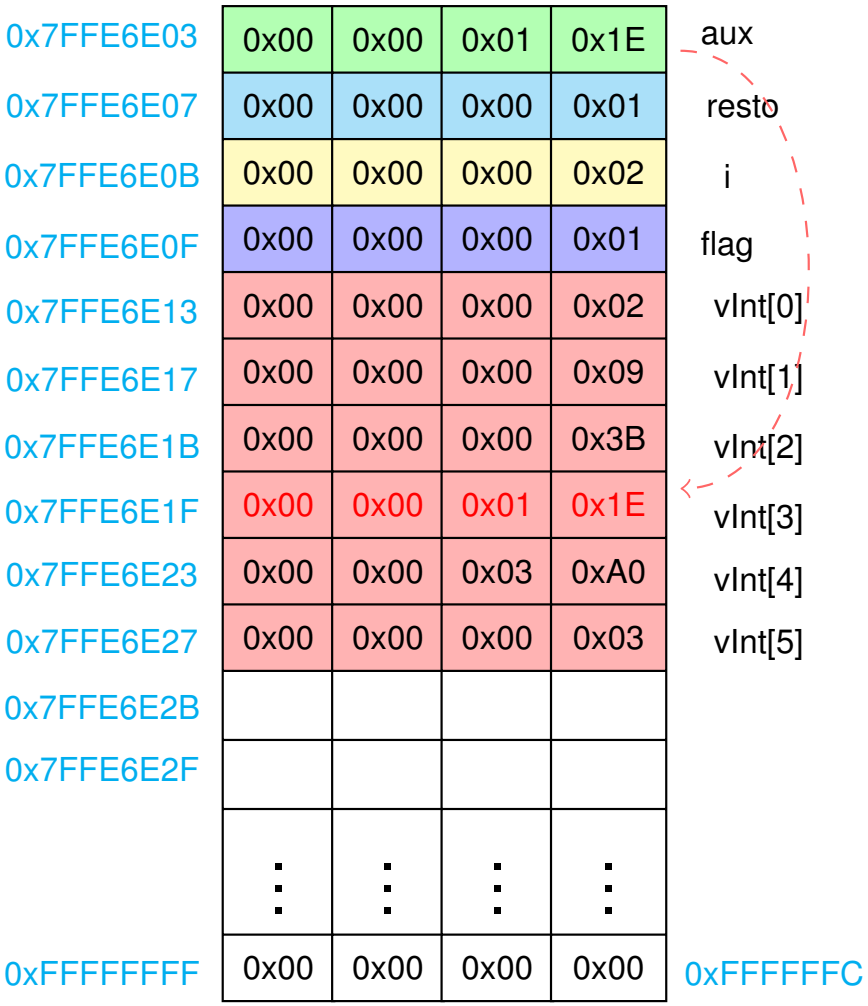
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

vInt[2] = vInt[3]

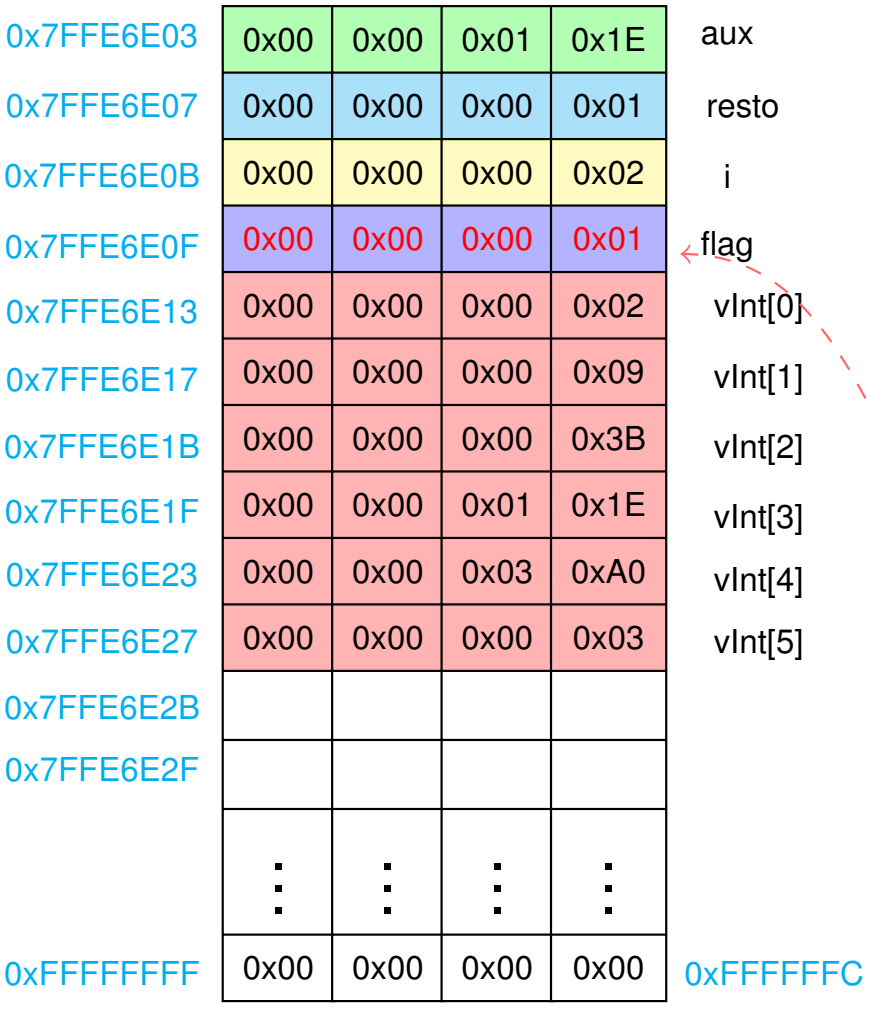
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

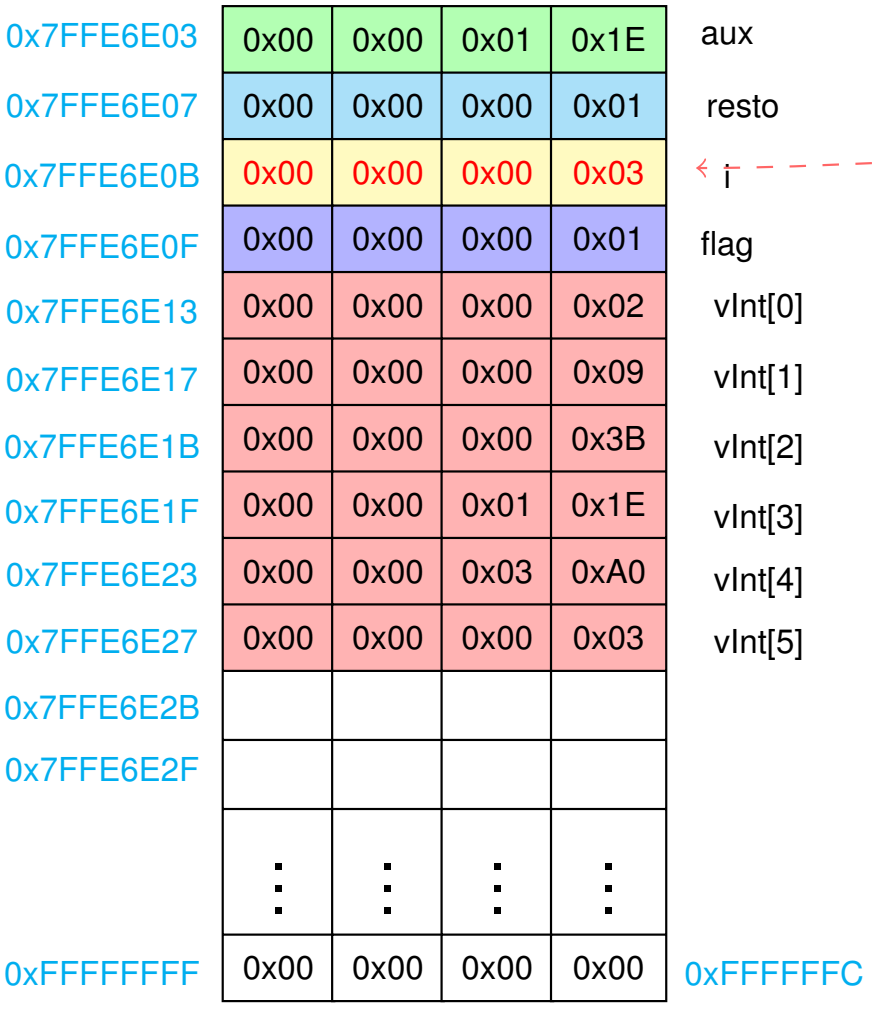
vInt[3] = aux

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

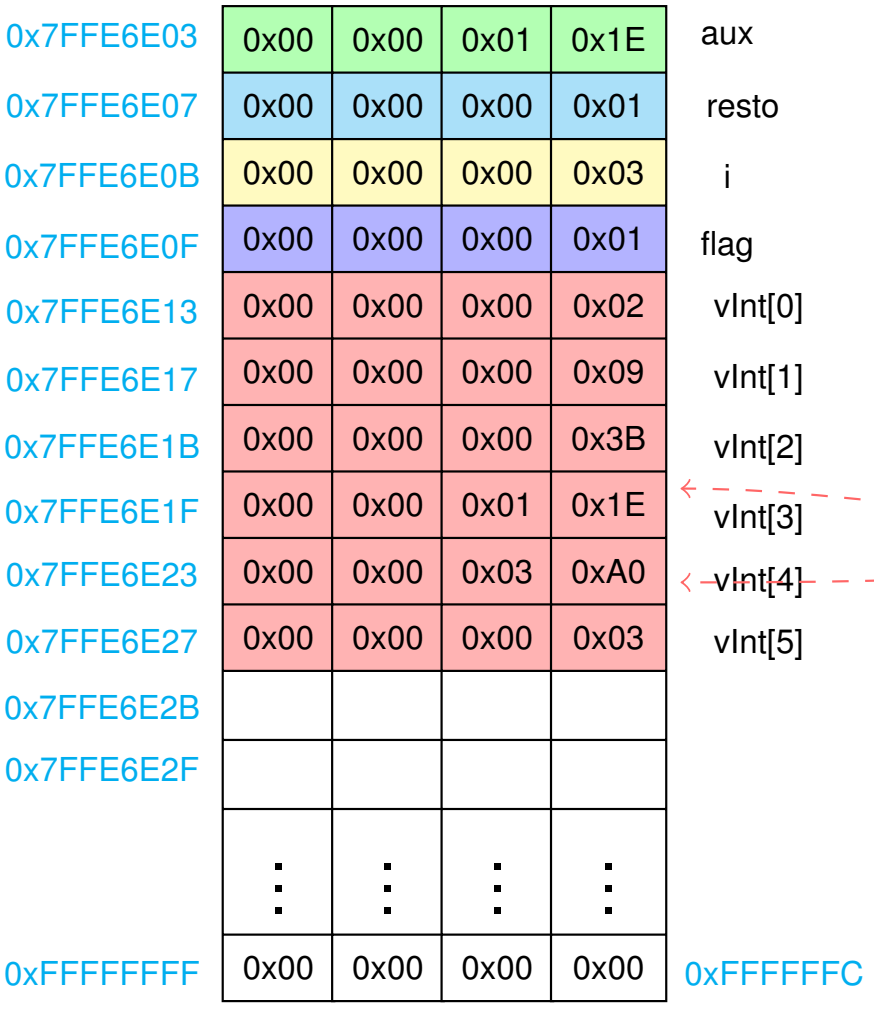
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

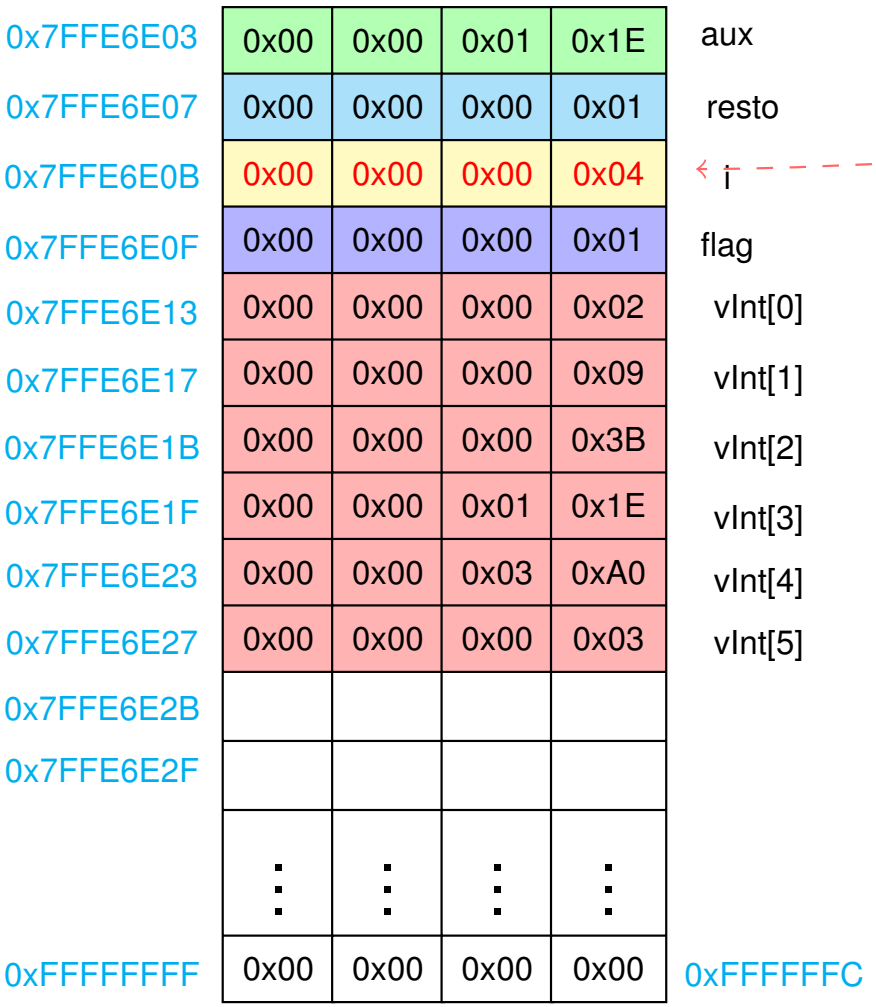
6-1=5

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {                  3              4
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

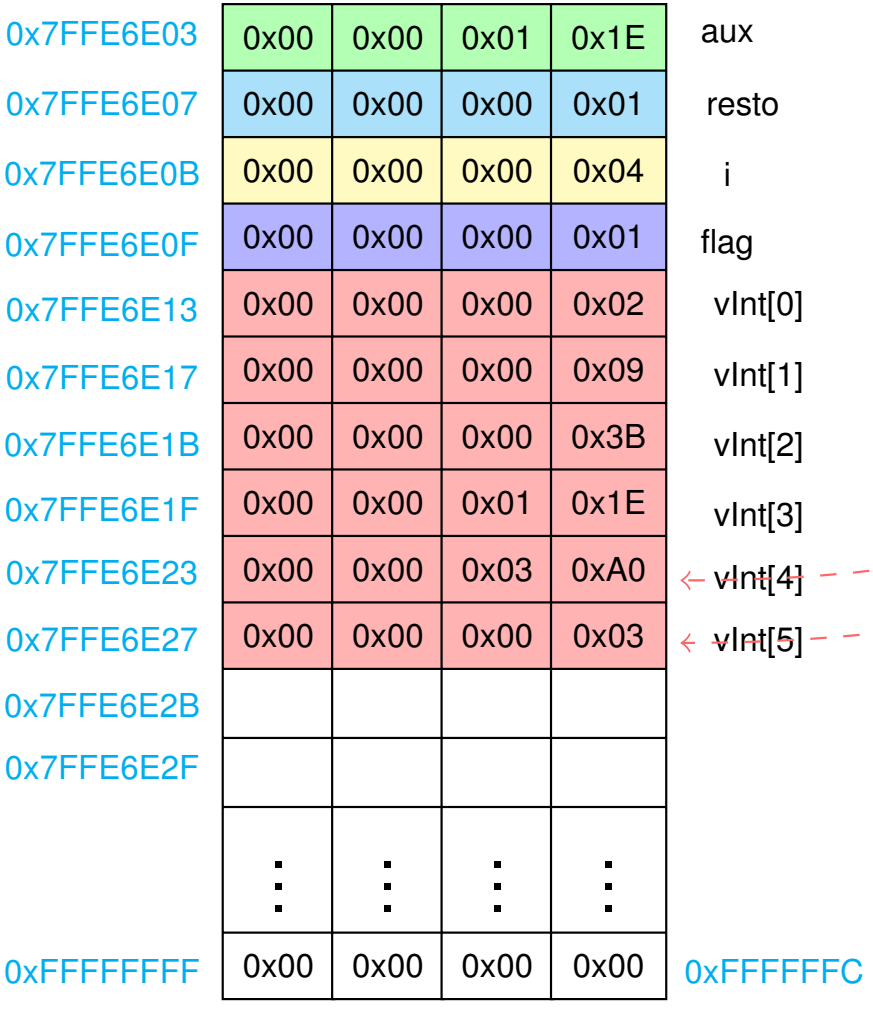
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

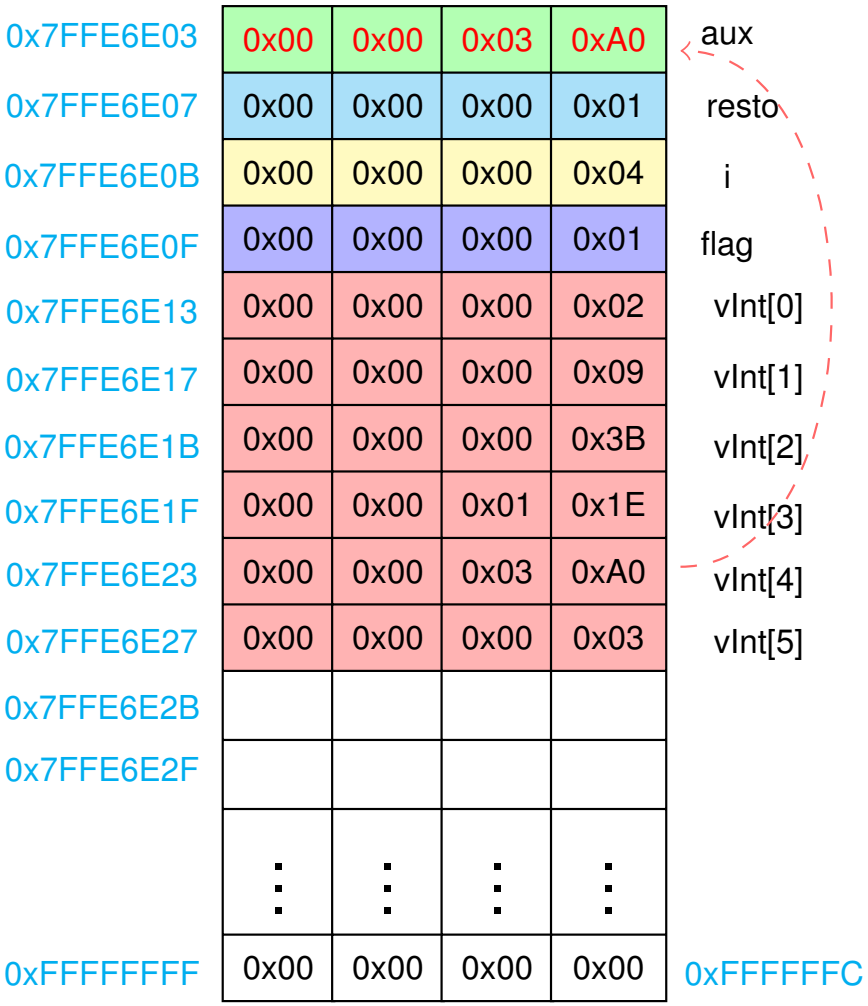
6-1=5

MEMORIA en ARQUITECTURA x86 32-bits



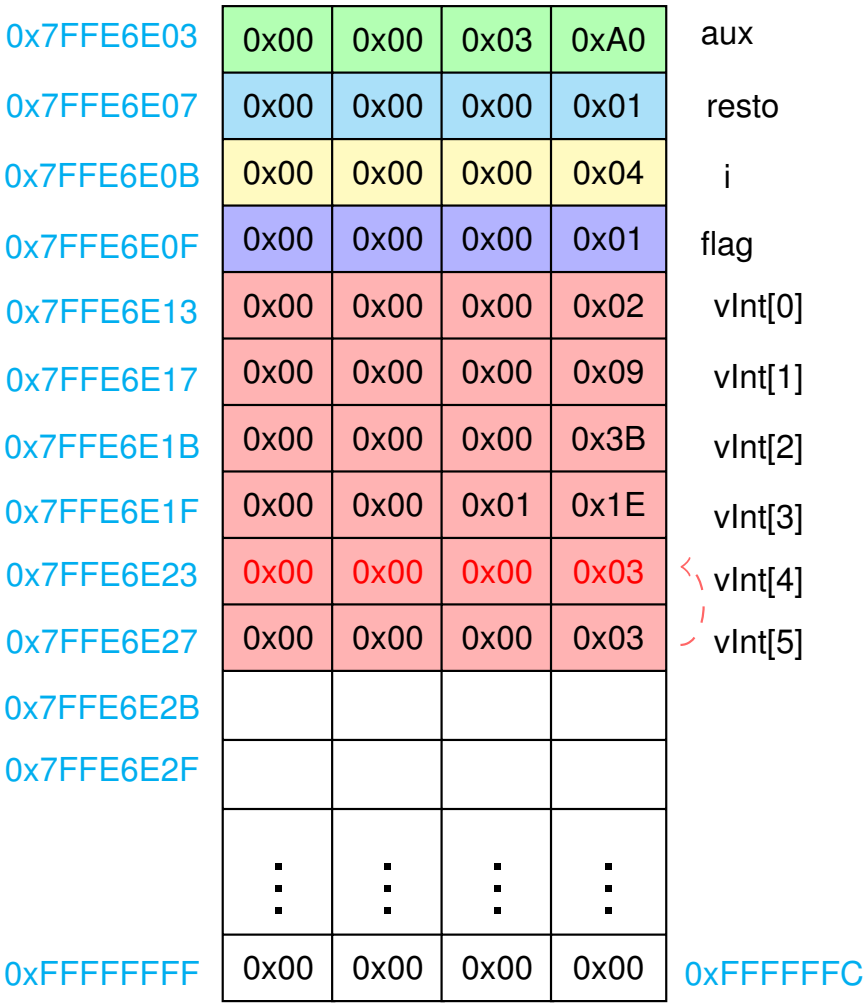

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {        4              5
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {                4
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits

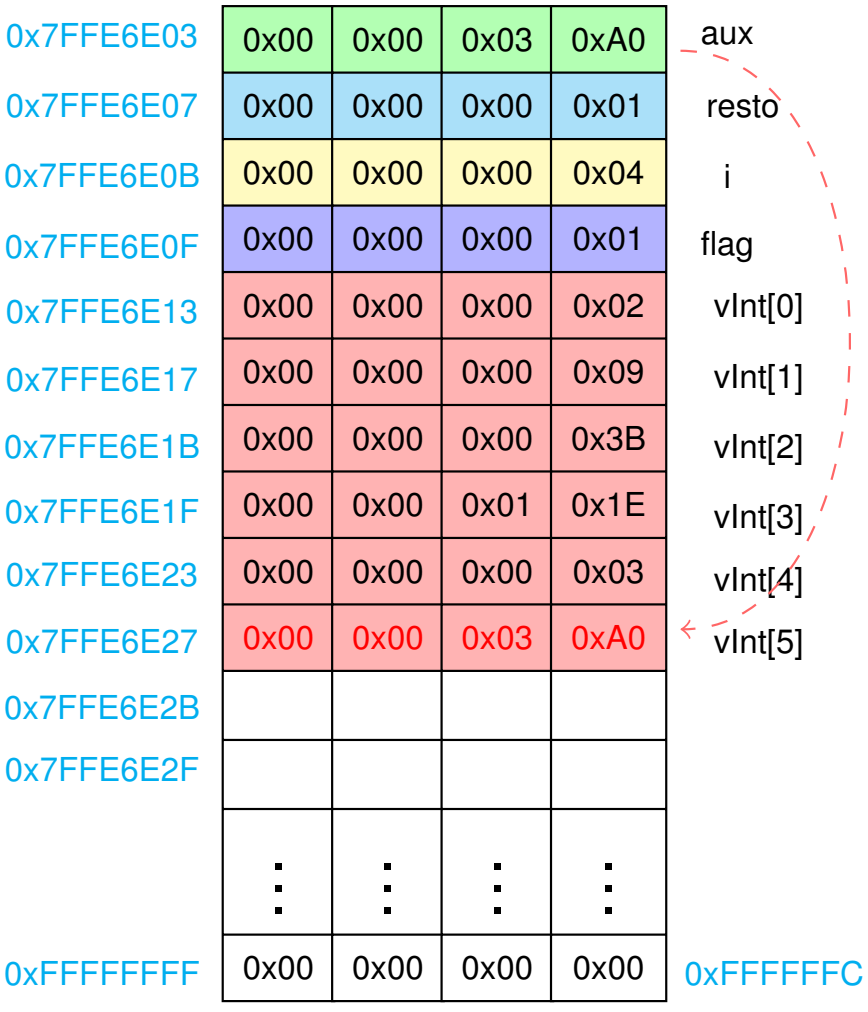



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

vInt[4] = vInt[5]

MEMORIA en ARQUITECTURA x86 32-bits

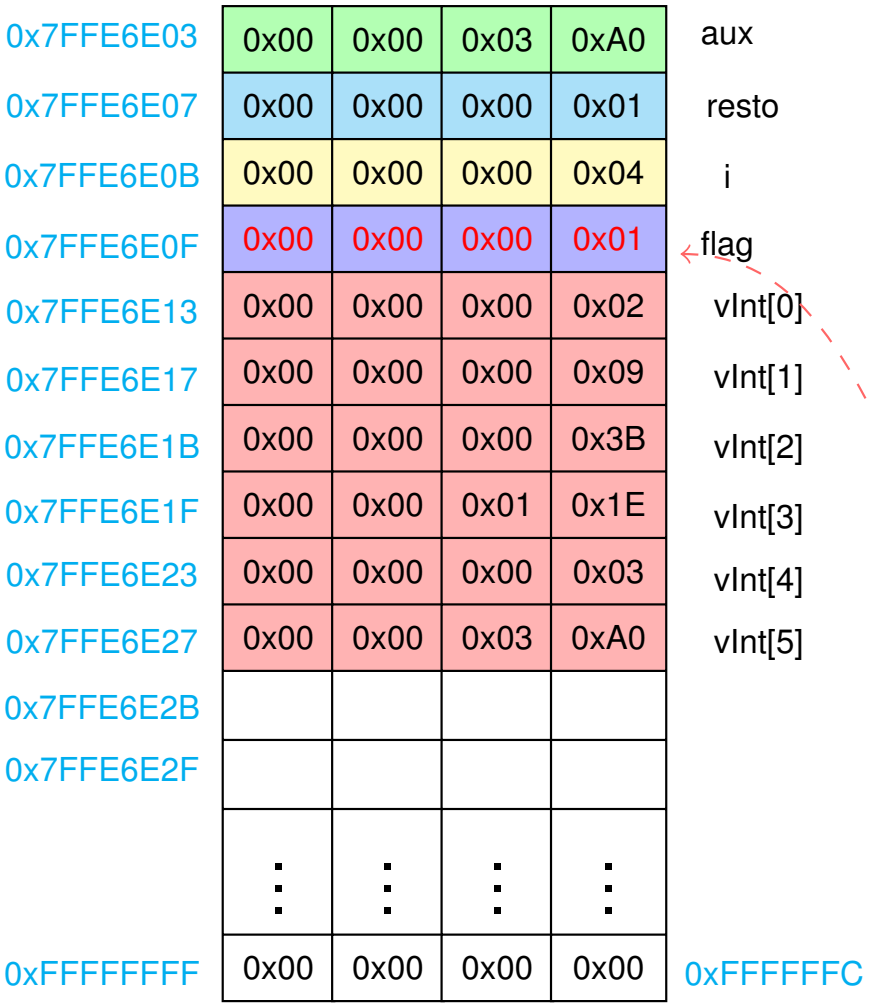



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

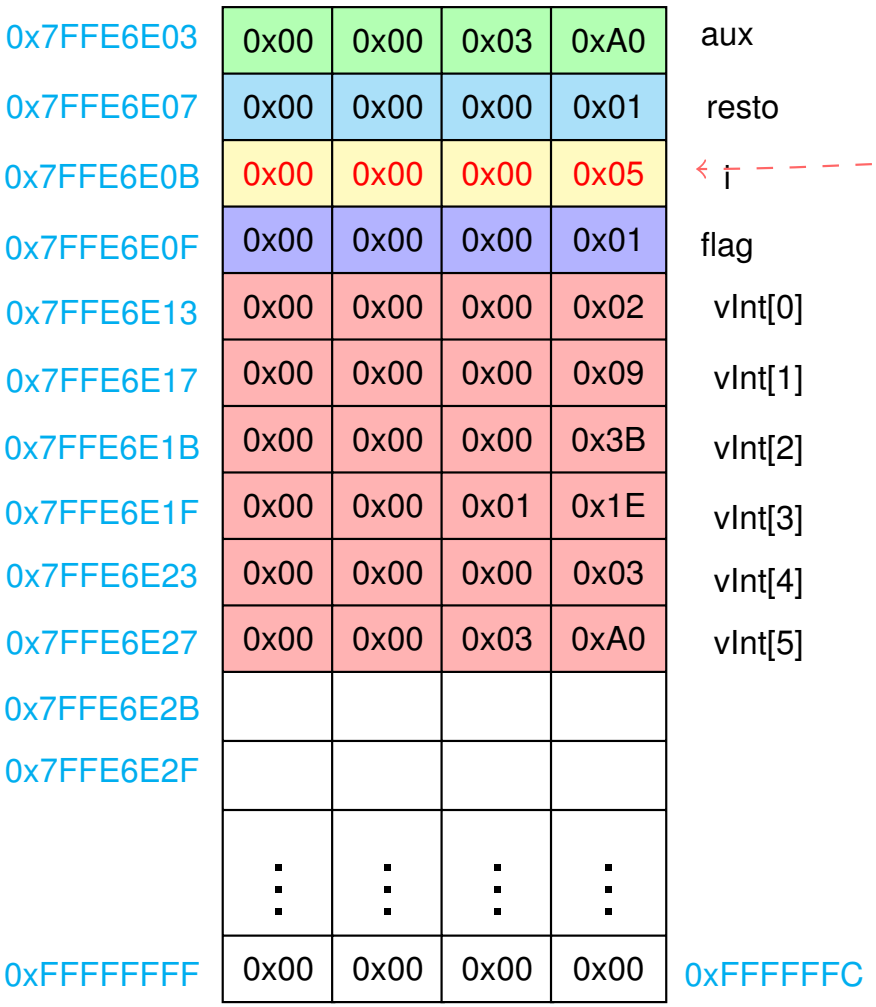
vInt[5] = aux

MEMORIA en ARQUITECTURA x86 32-bits



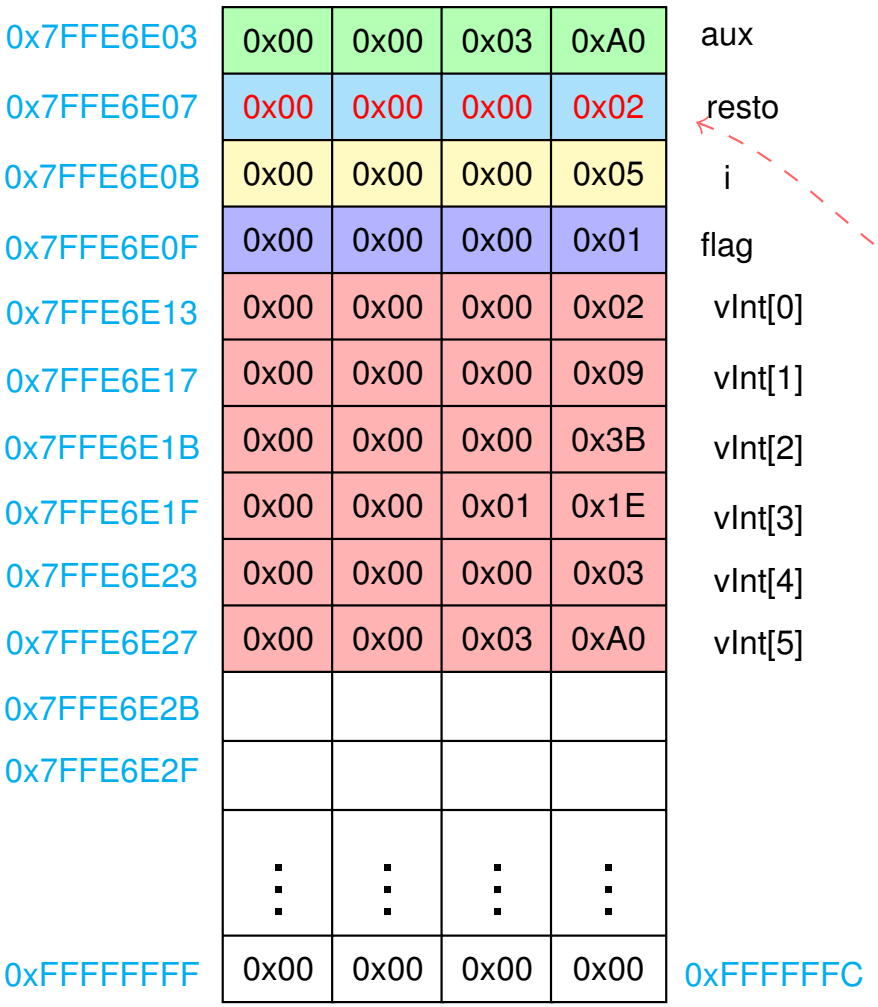

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits



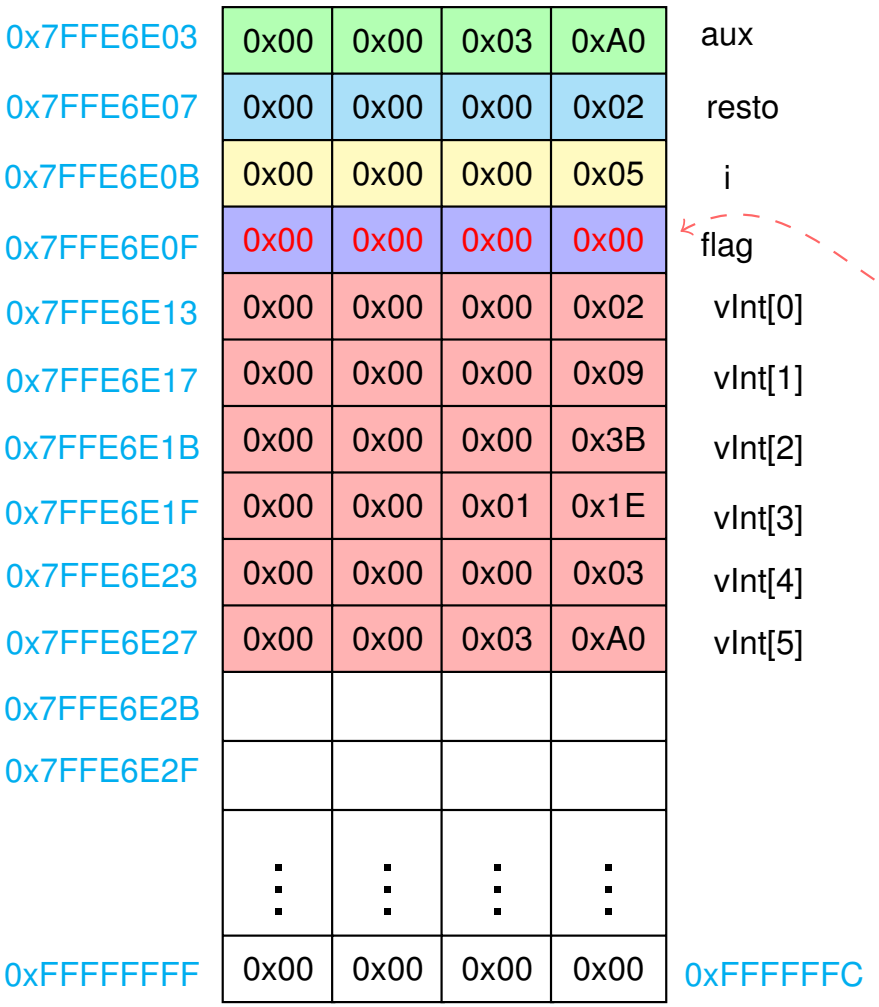

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-1=5
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

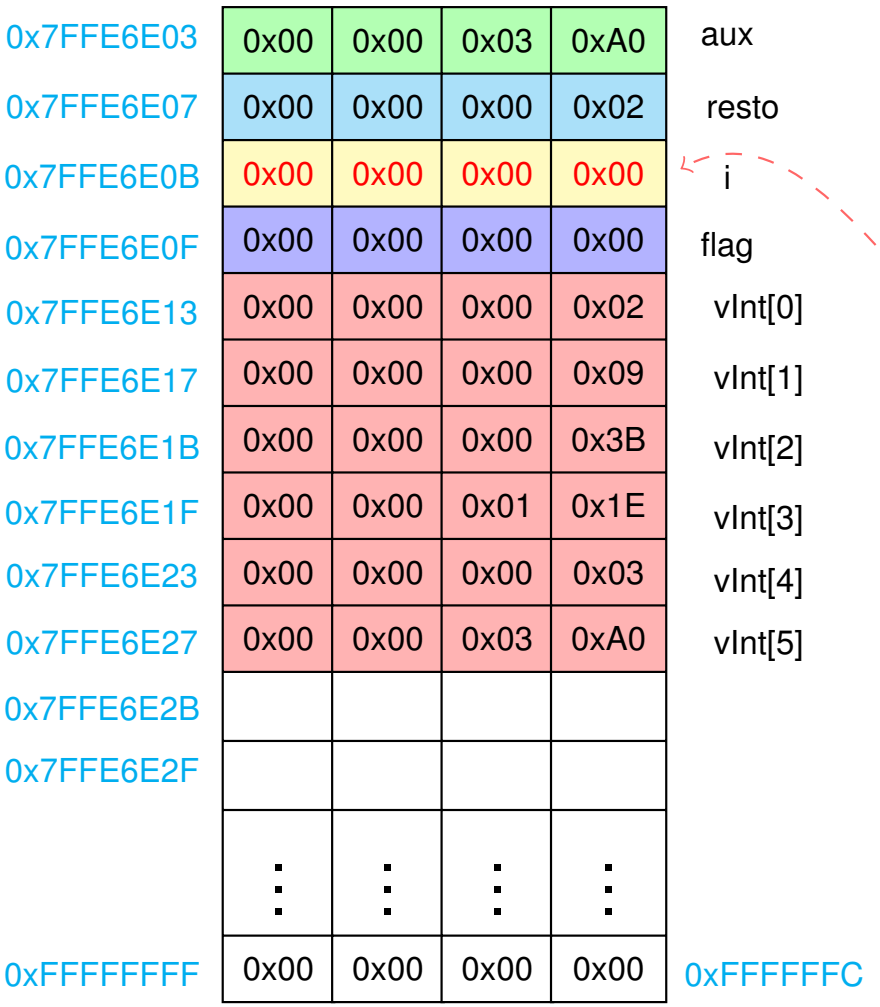
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-1=5

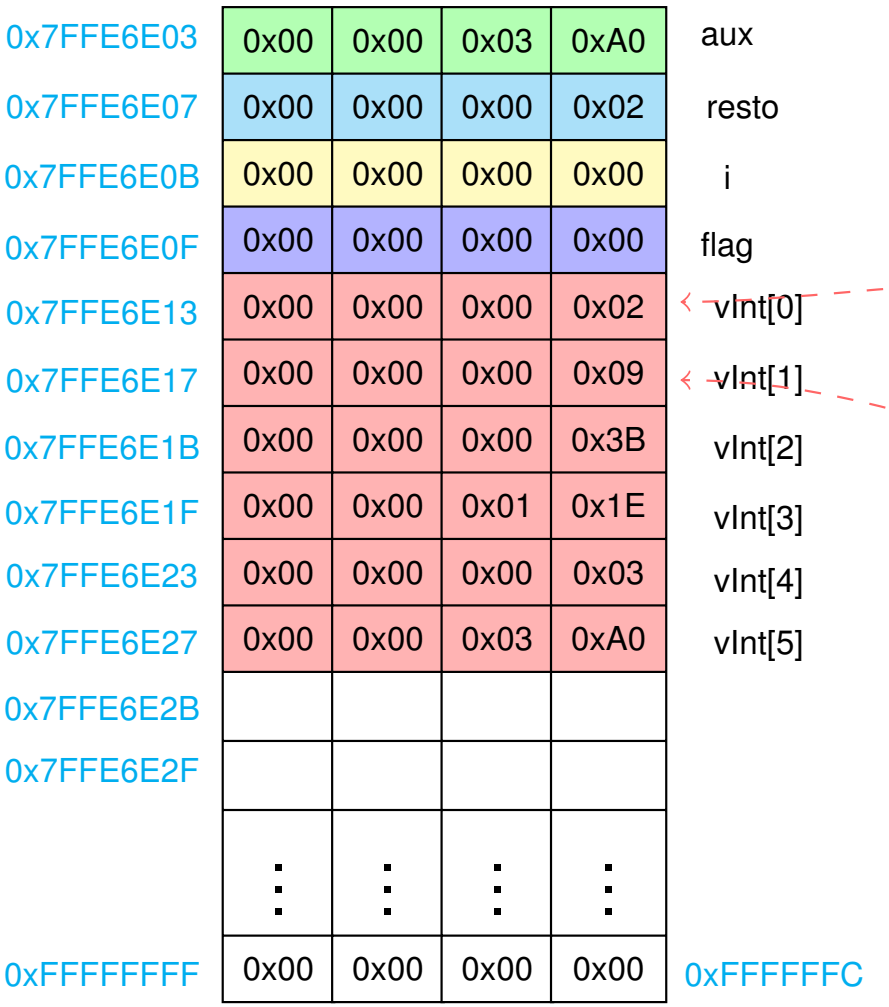
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

MEMORIA en ARQUITECTURA x86 32-bits

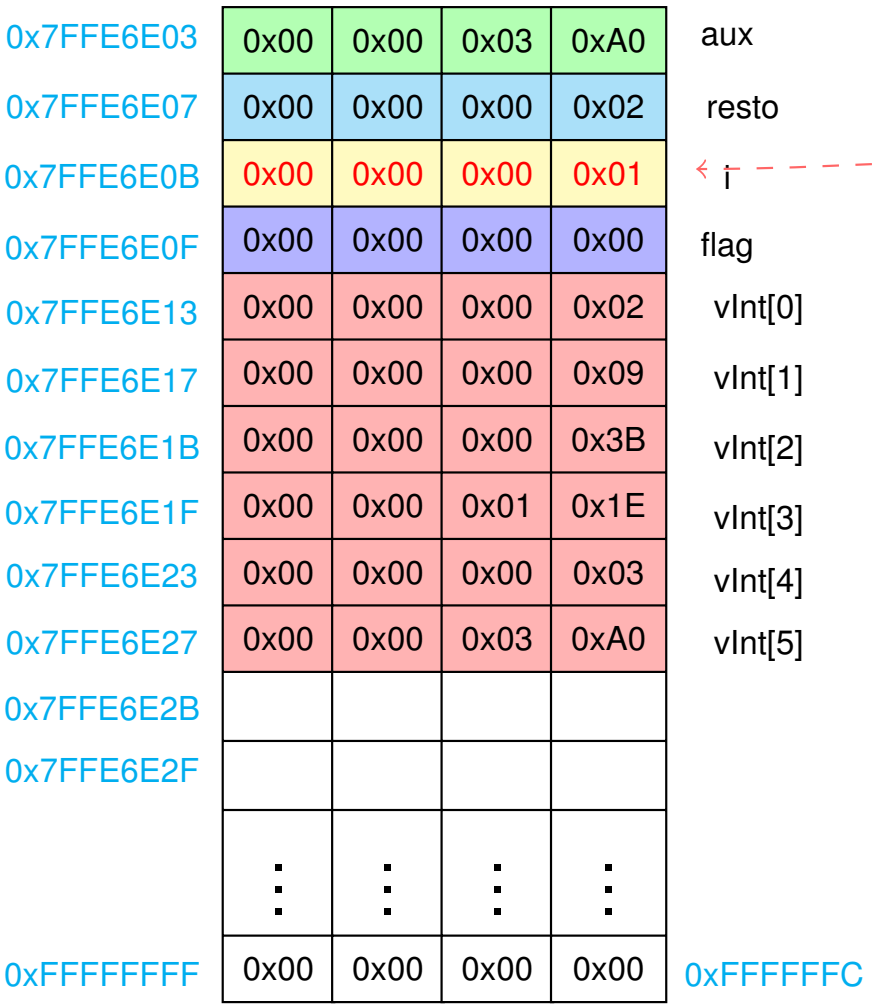



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

0 1

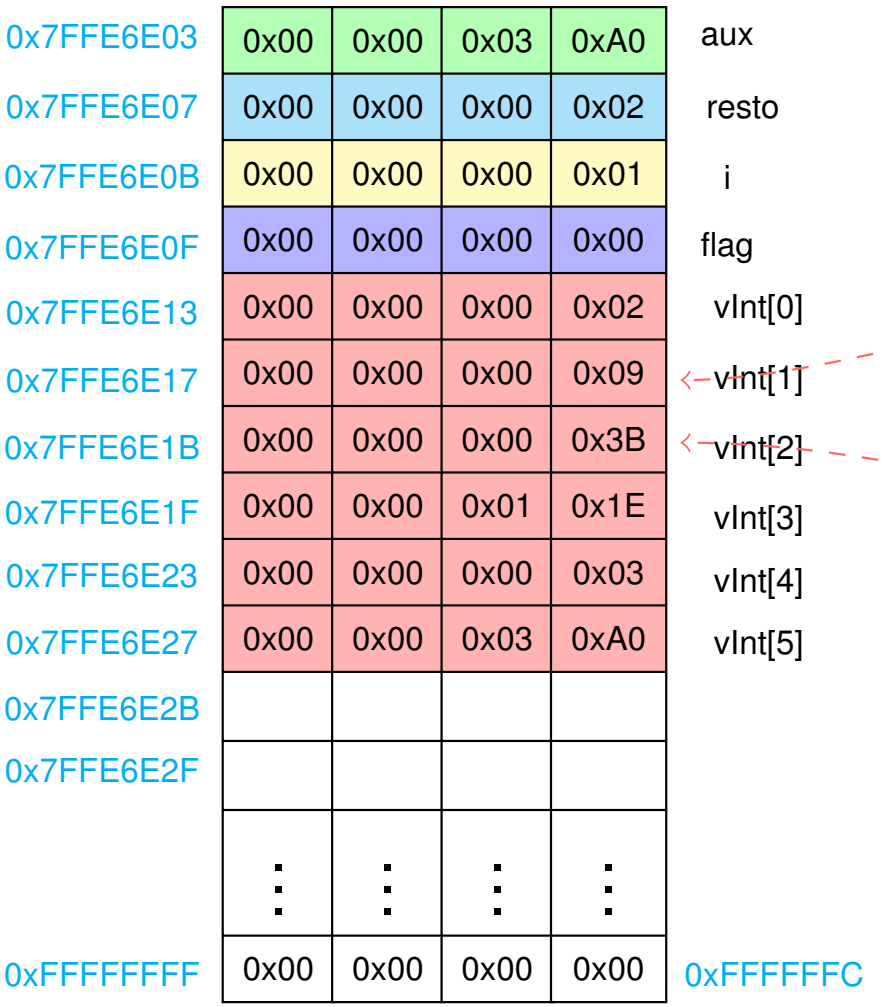
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

MEMORIA en ARQUITECTURA x86 32-bits

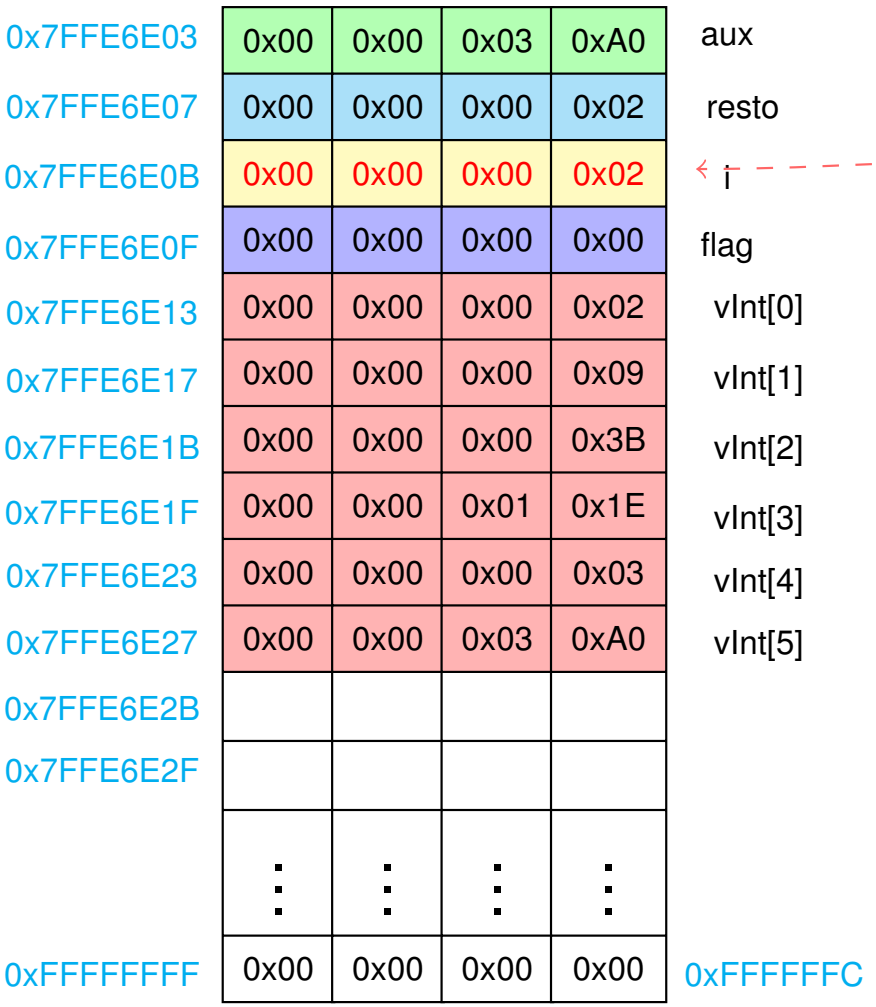



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

1 2

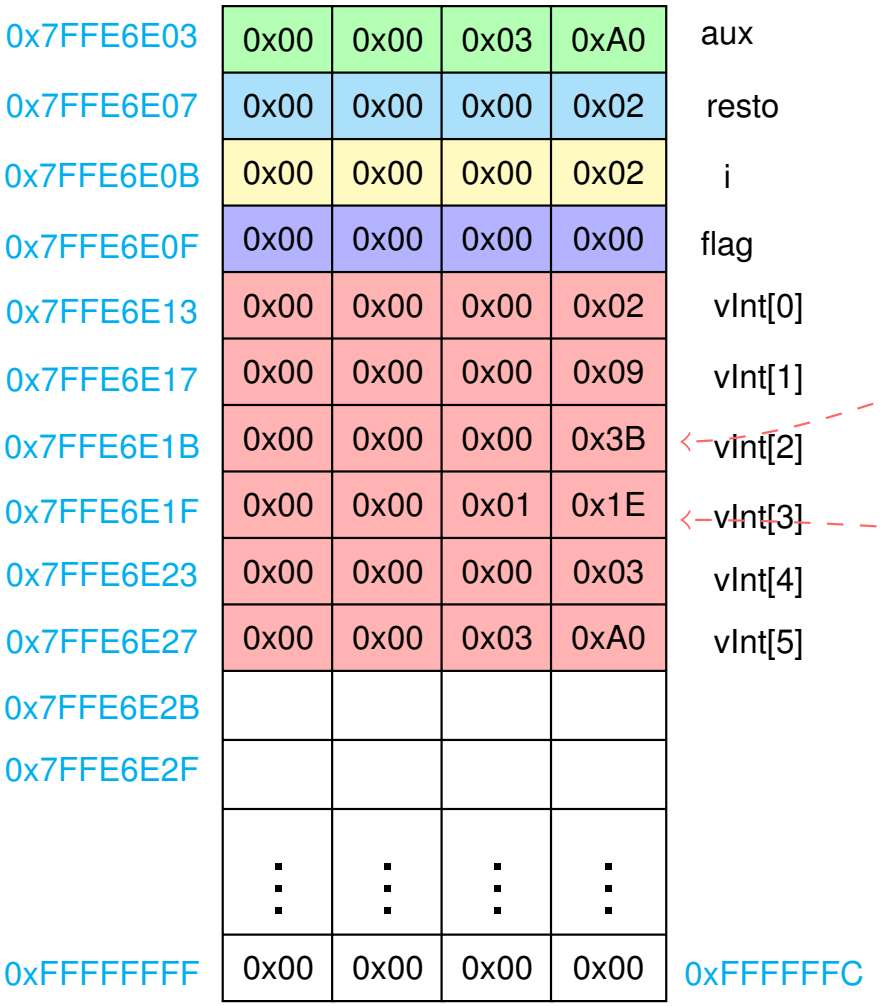
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

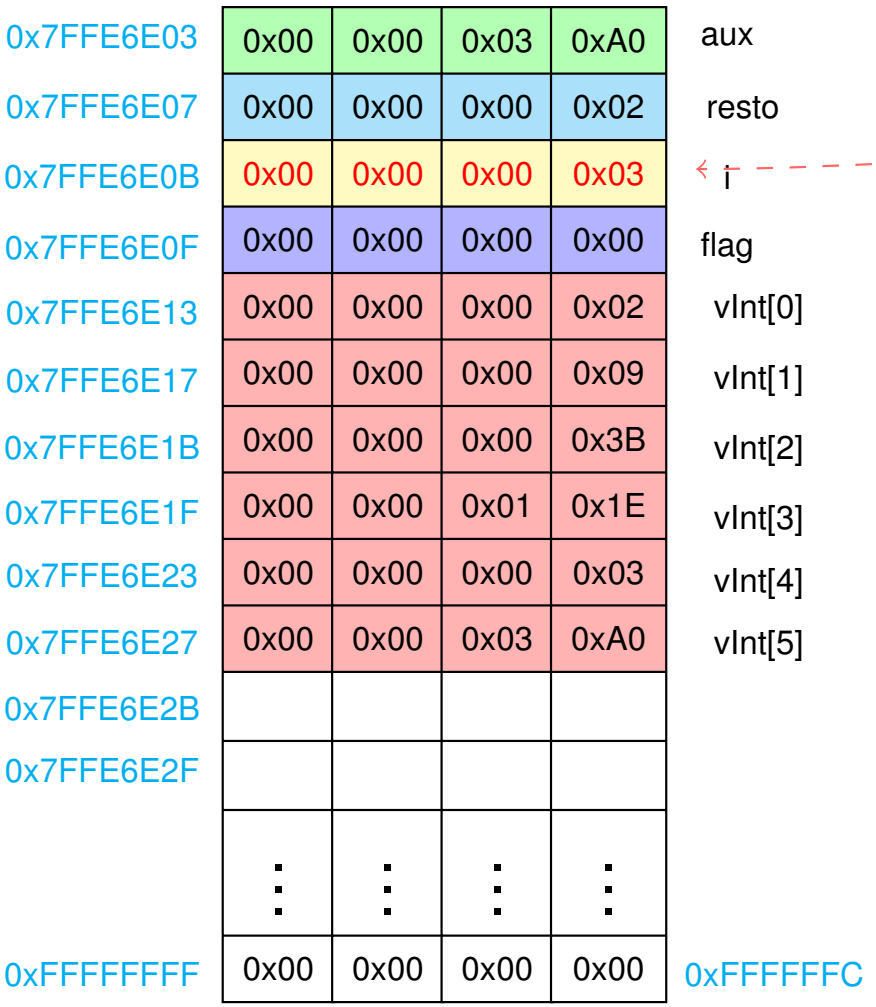
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

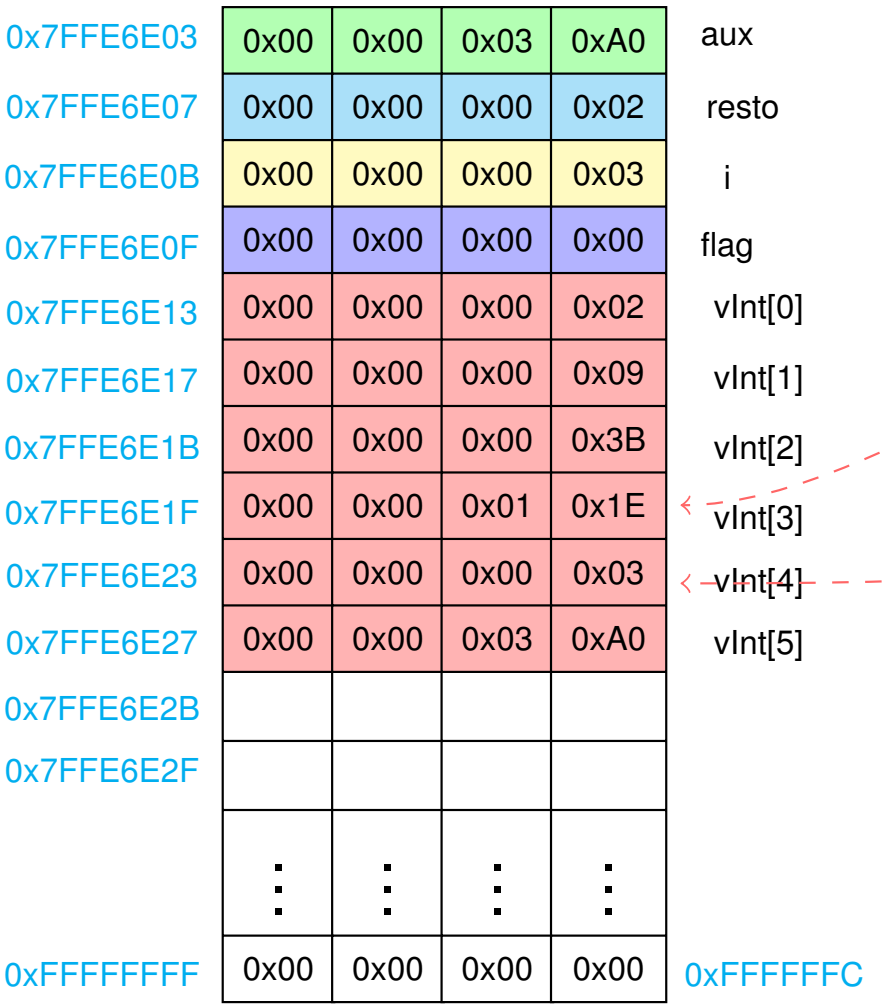
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

MEMORIA en ARQUITECTURA x86 32-bits



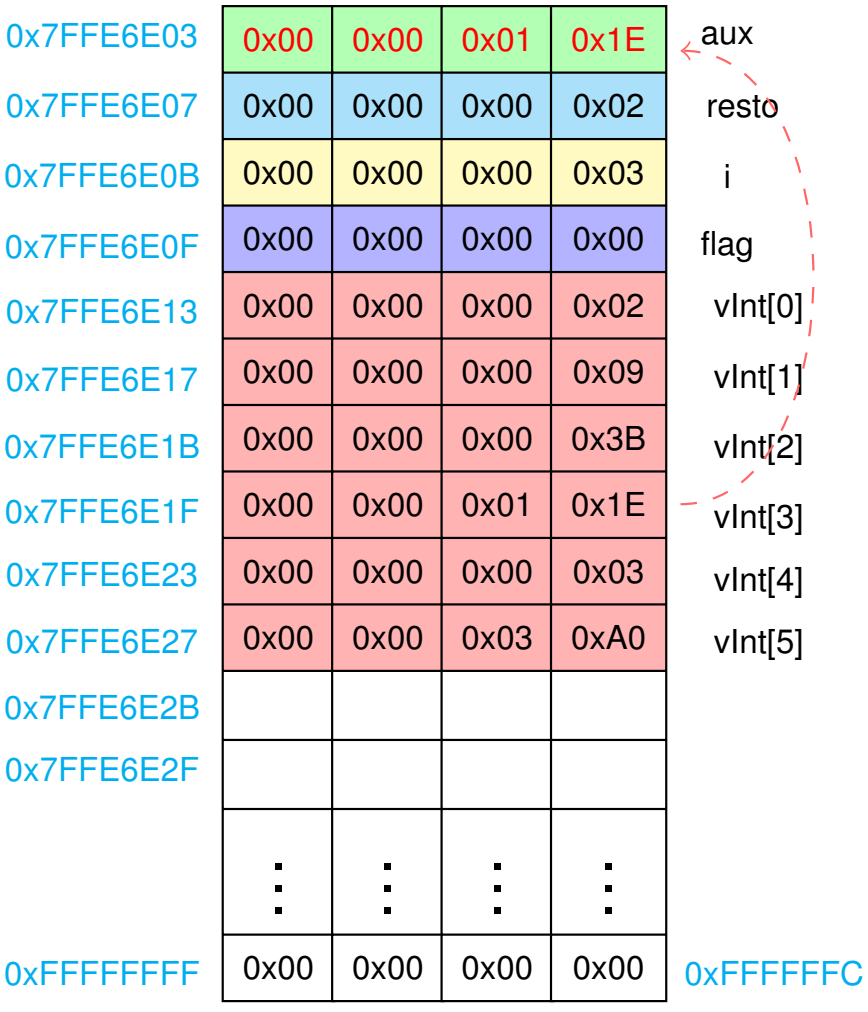

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

3

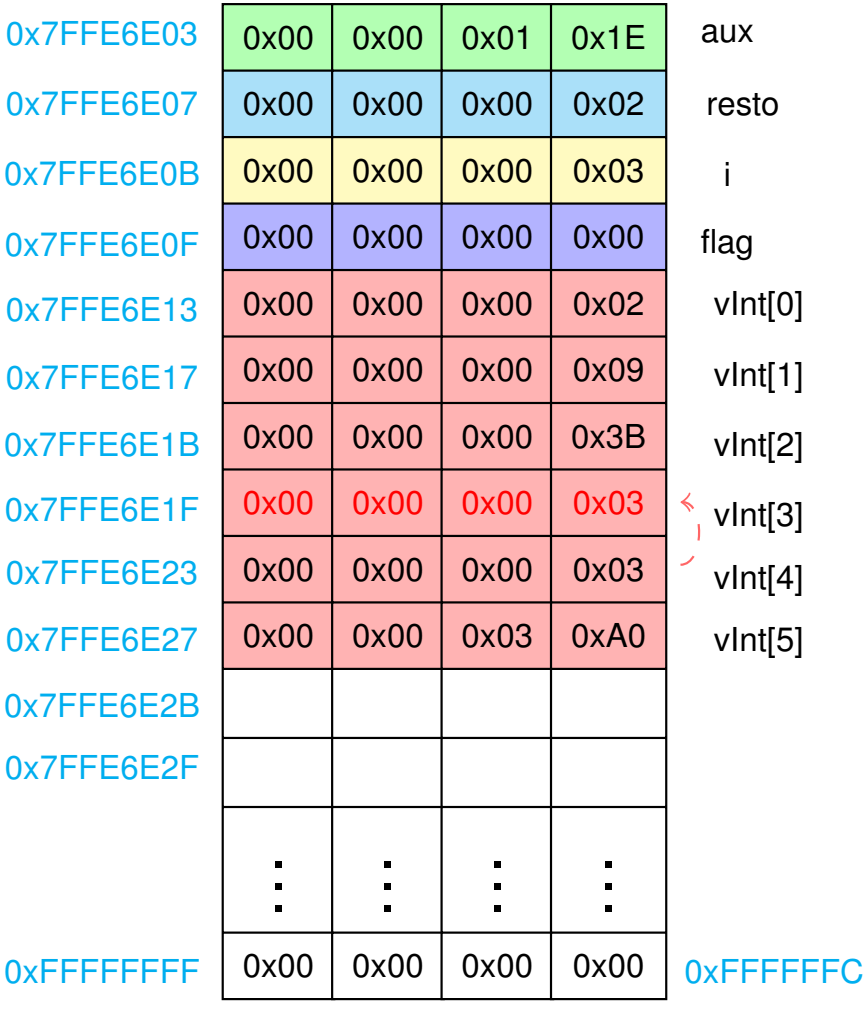
4

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-2=4
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {              3
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits

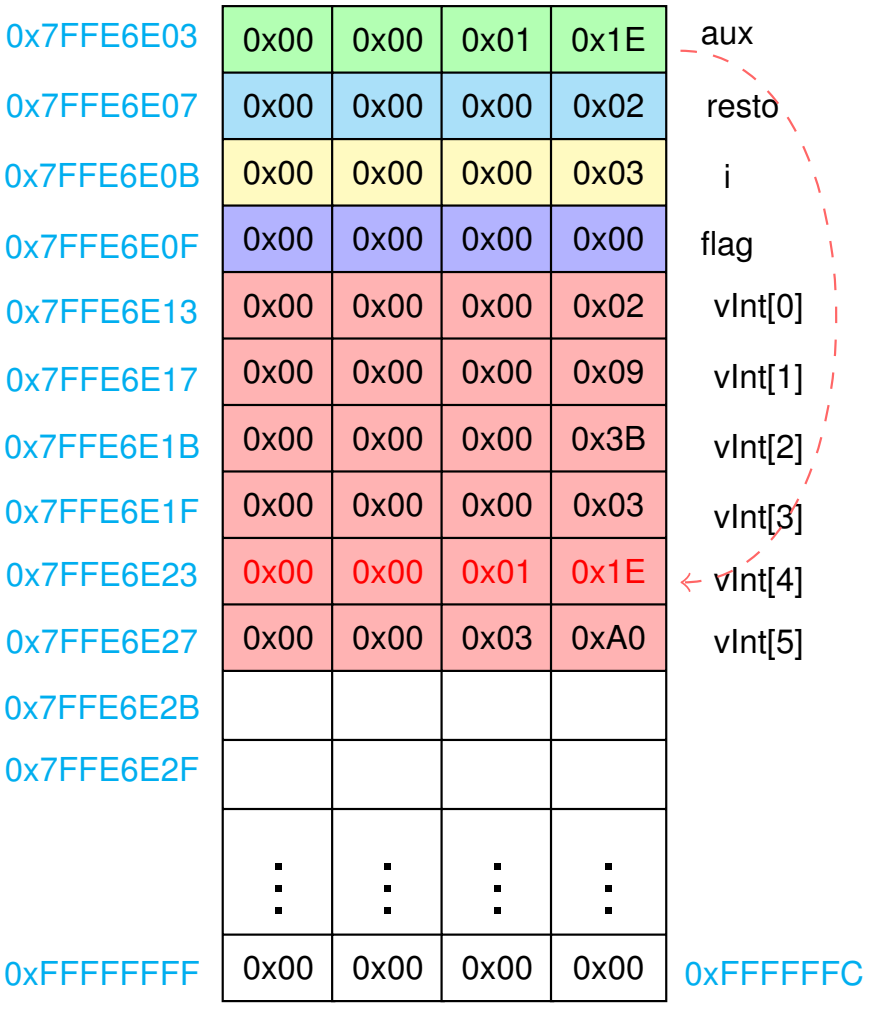



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

vInt[3] = vInt[4]

MEMORIA en ARQUITECTURA x86 32-bits

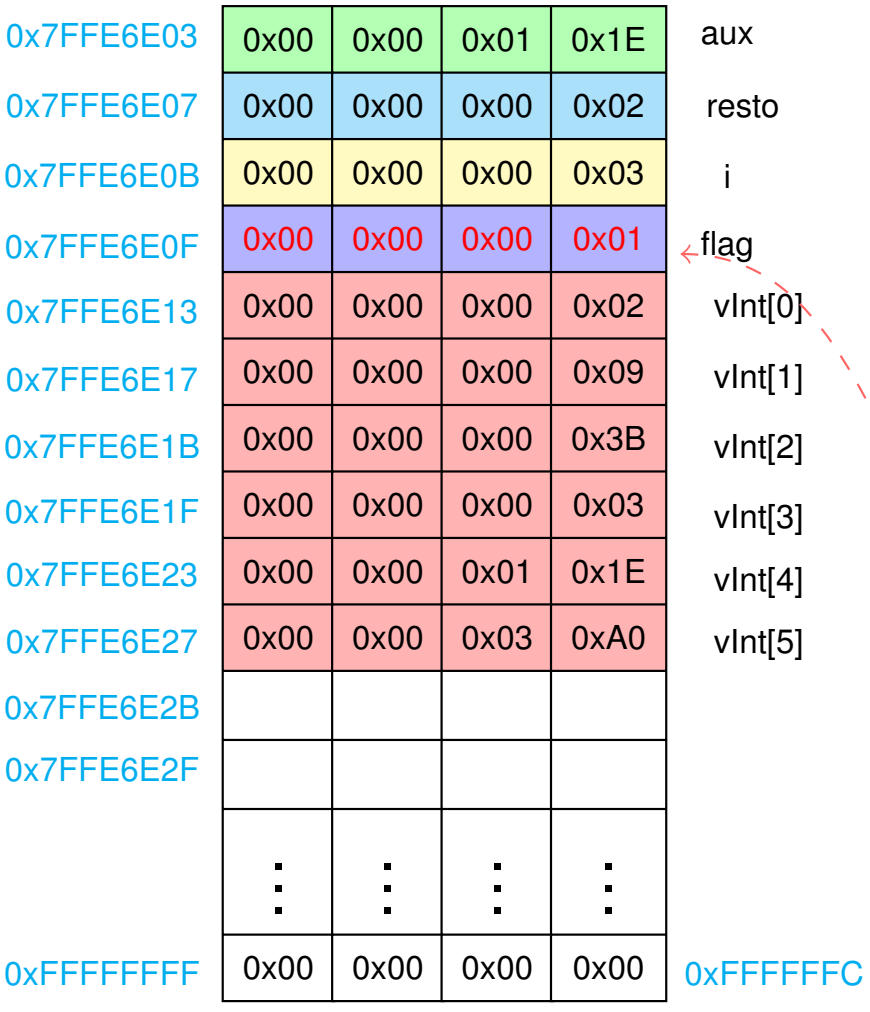



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

vInt[4] = aux

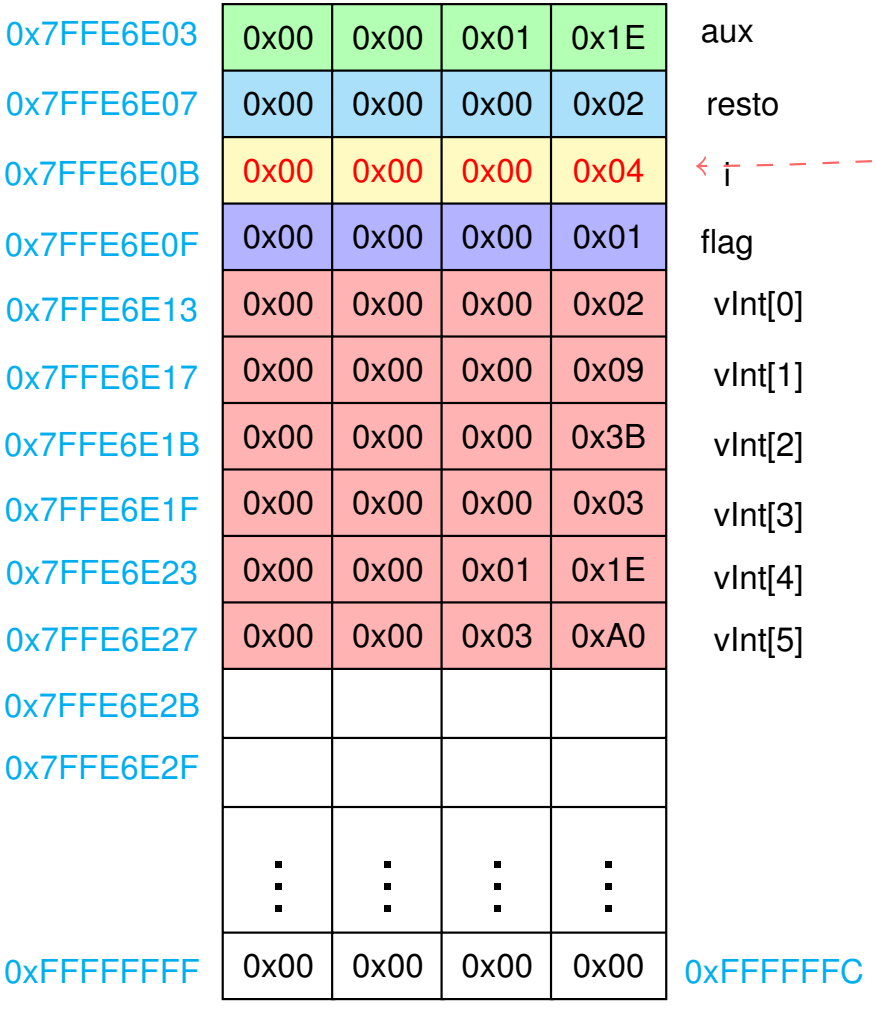
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

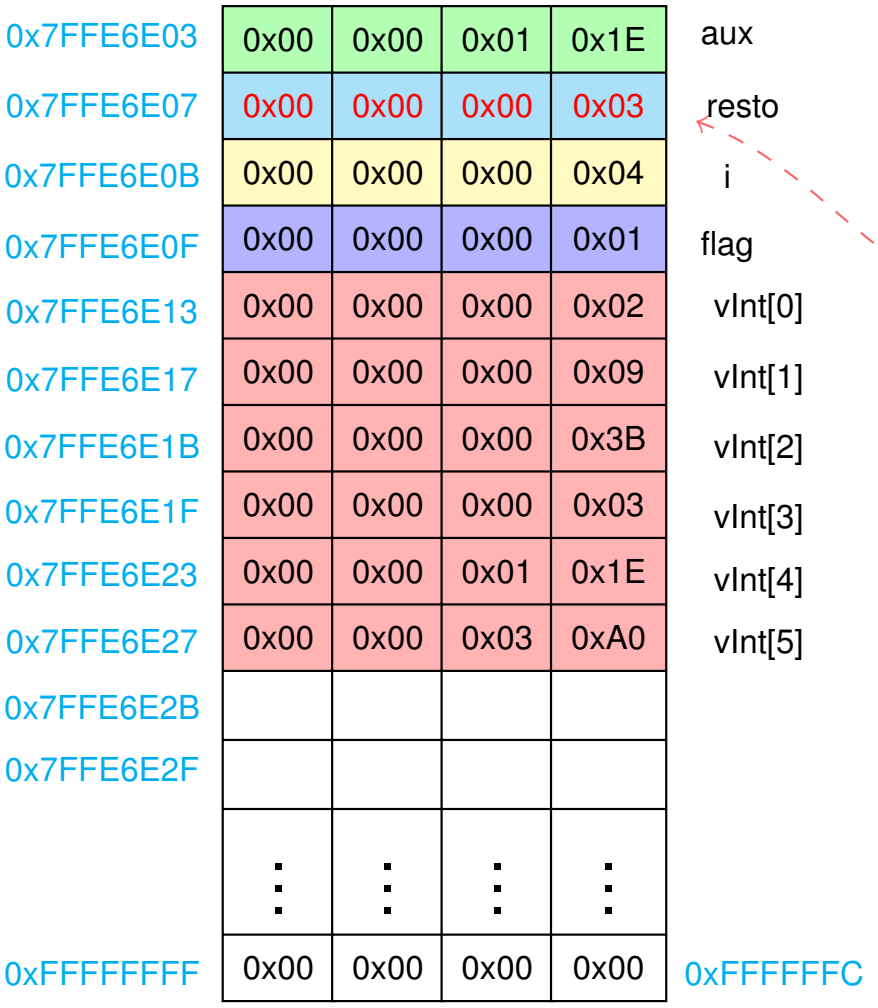
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

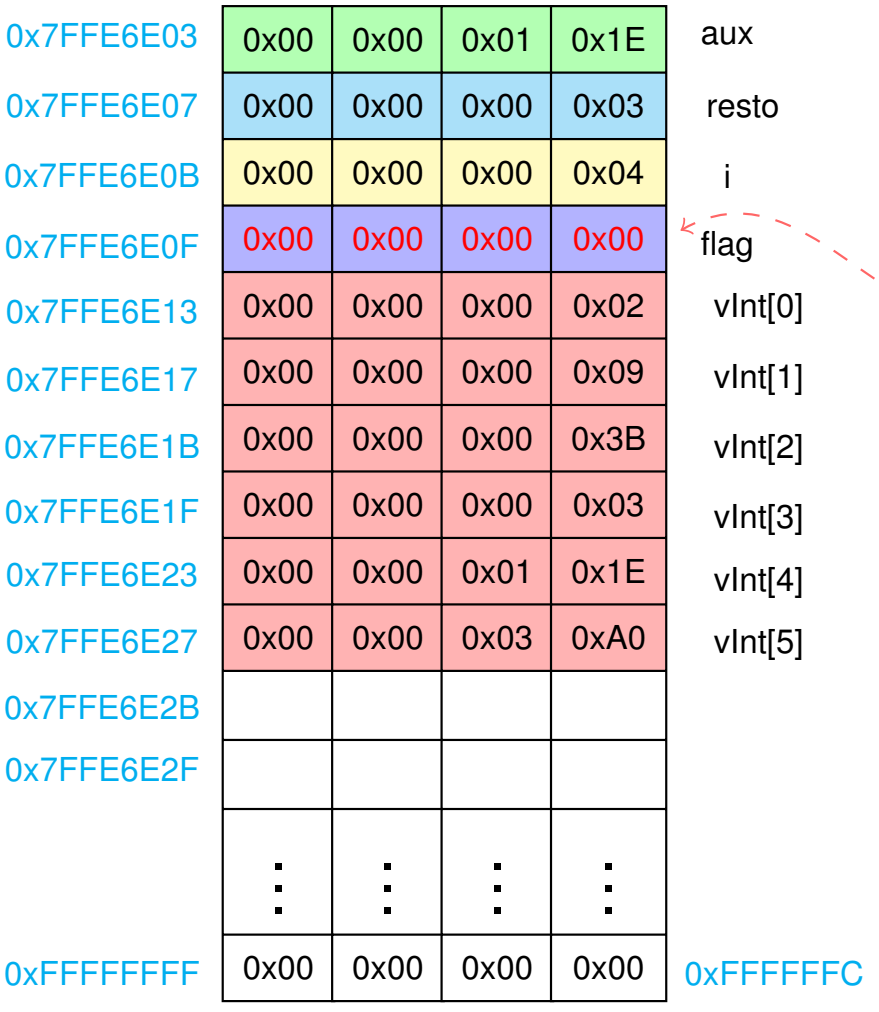
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

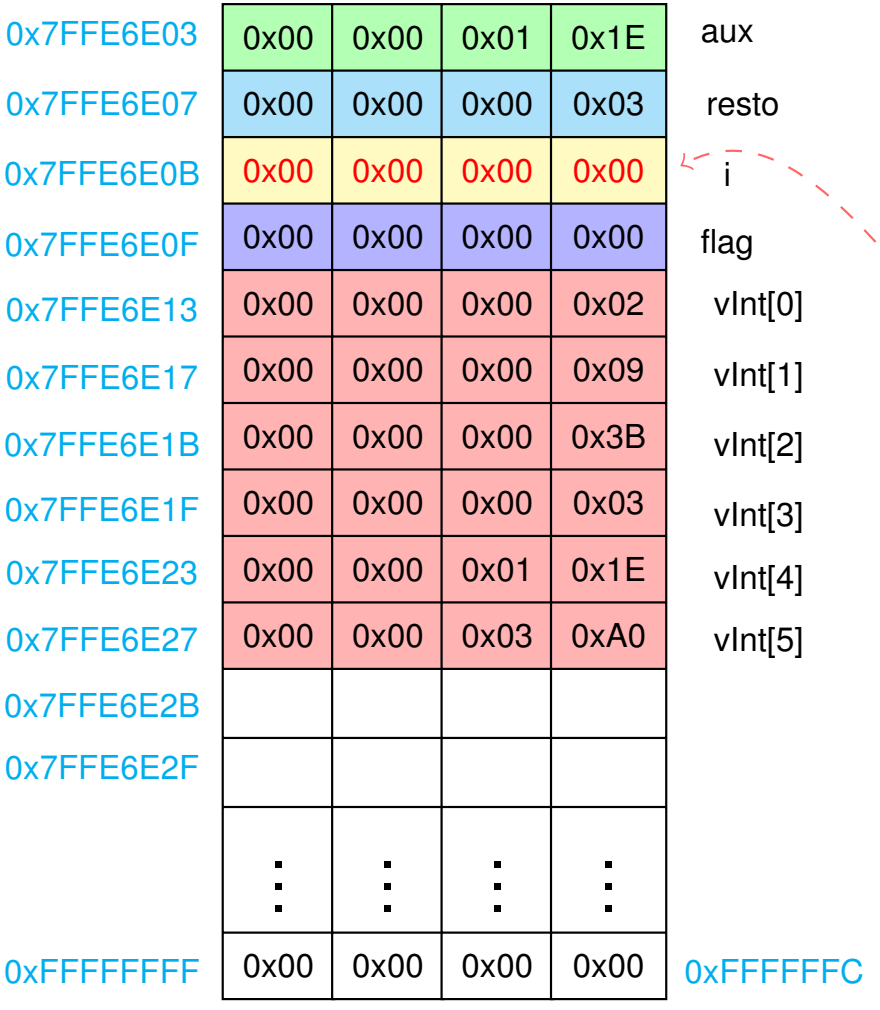
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-2=4

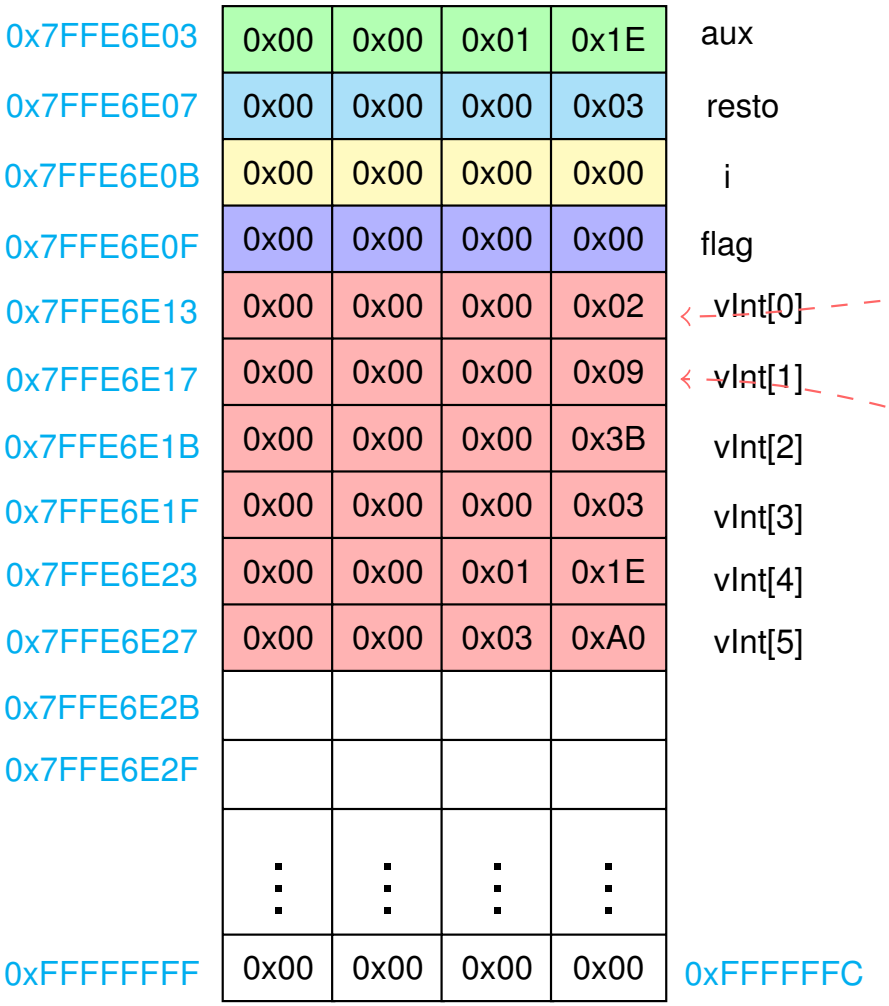
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

MEMORIA en ARQUITECTURA x86 32-bits

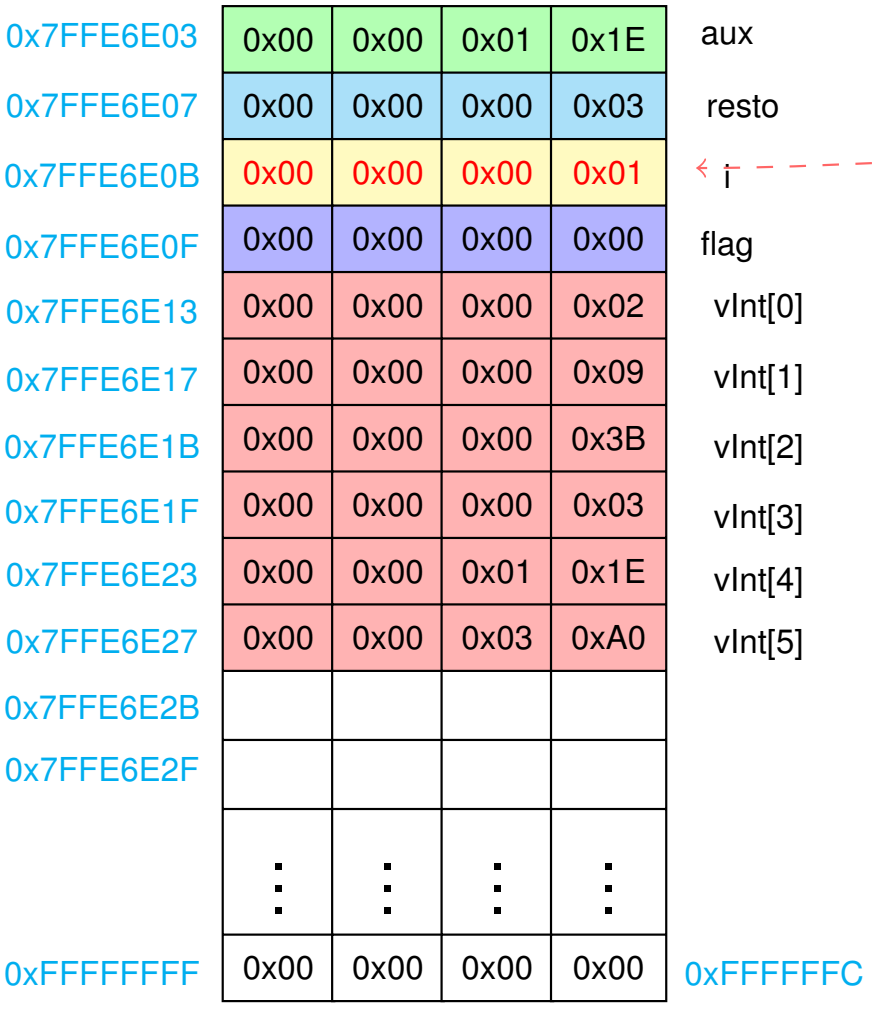



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

0 1

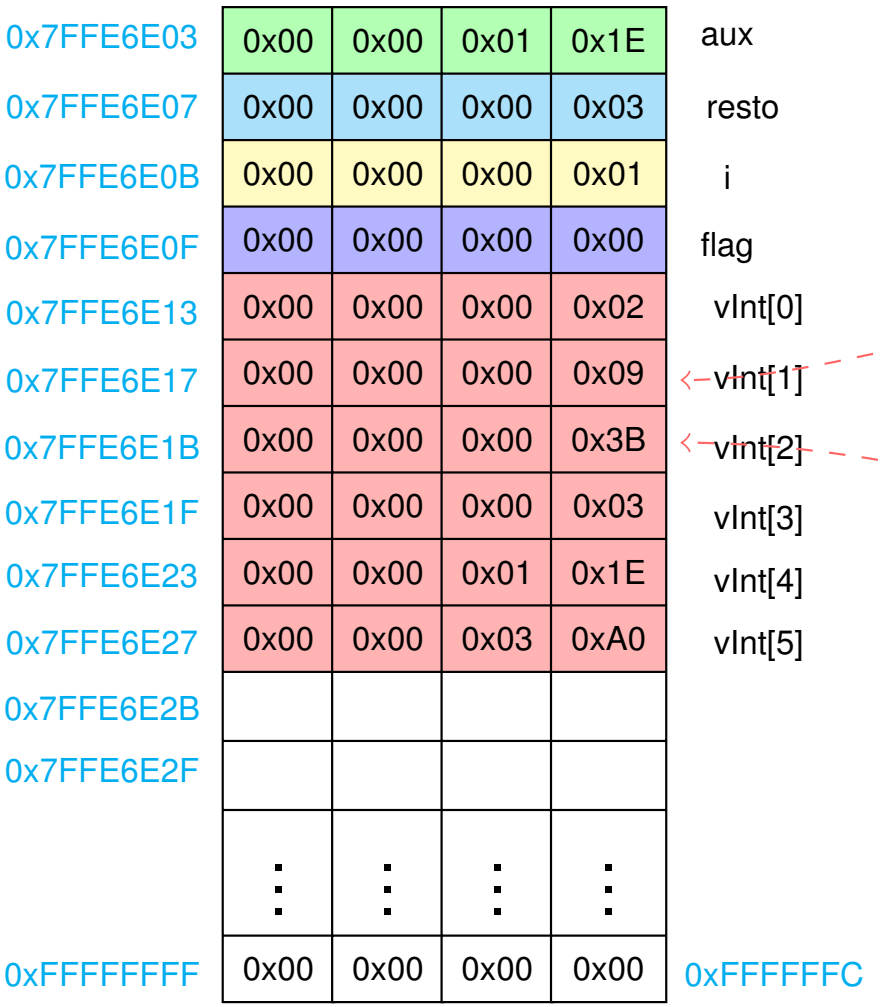
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

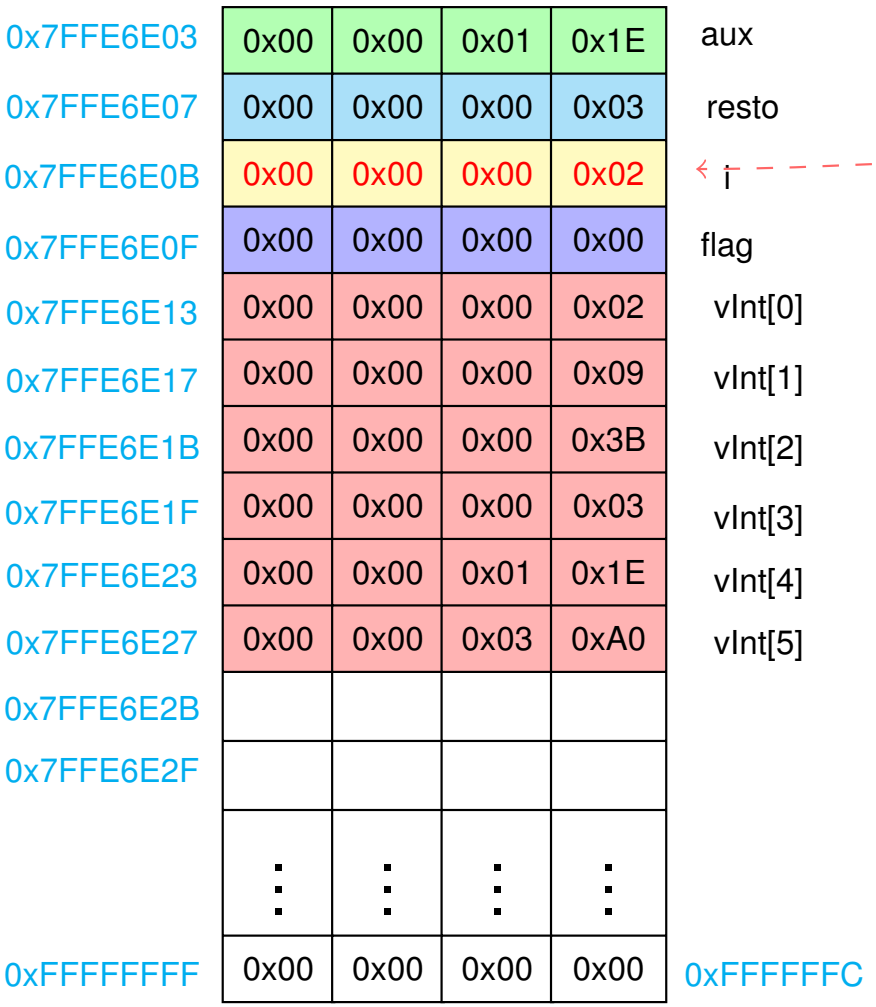
6-3=3

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-3=3
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {          1              2
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

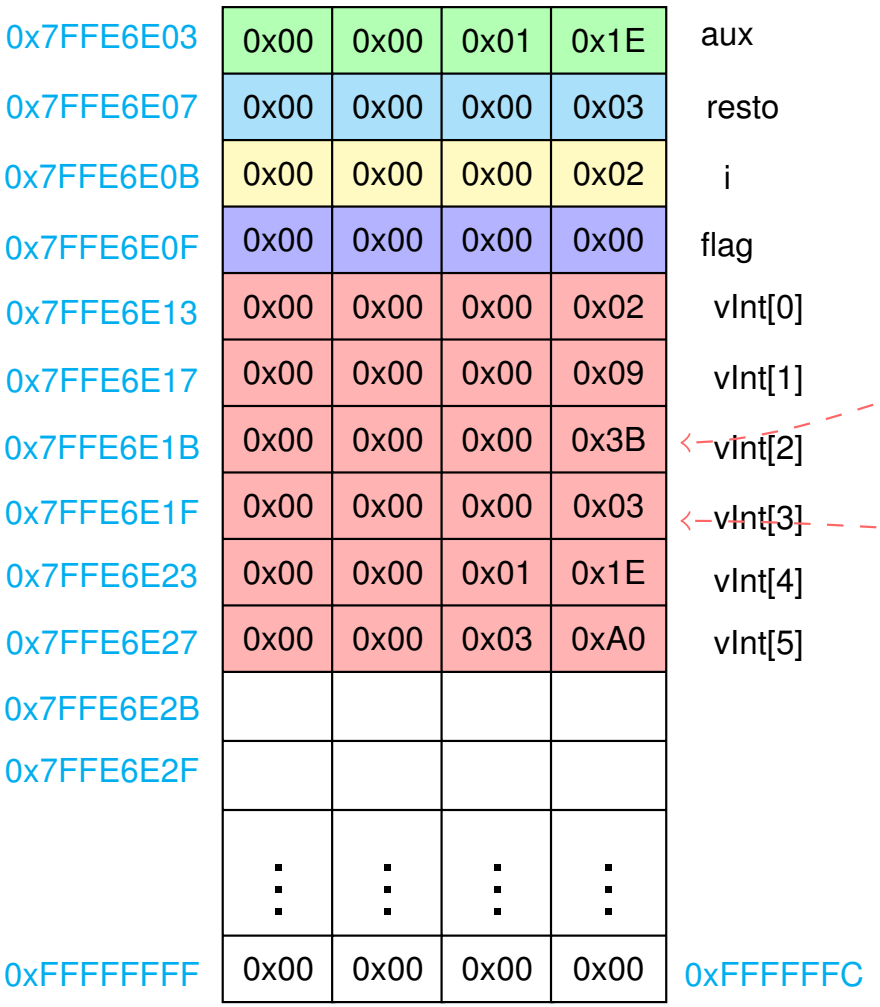
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

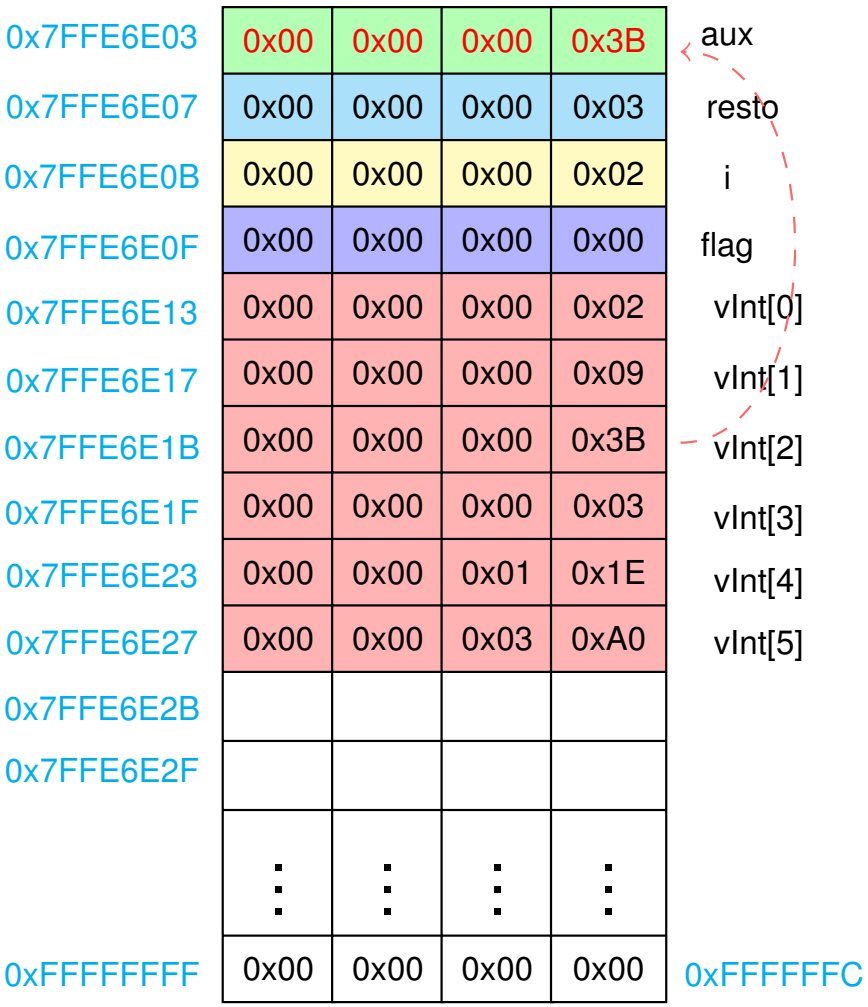
6-3=3

MEMORIA en ARQUITECTURA x86 32-bits



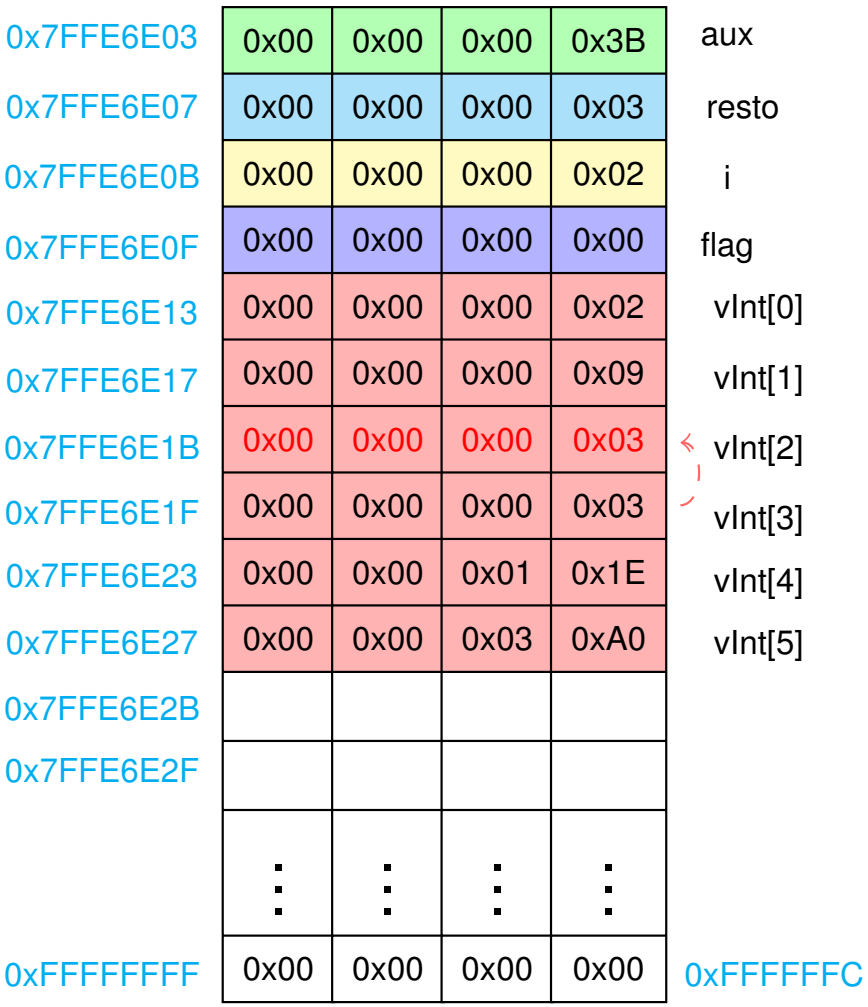

```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-3=3
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {          2                3
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                6-3=3
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {                2
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits

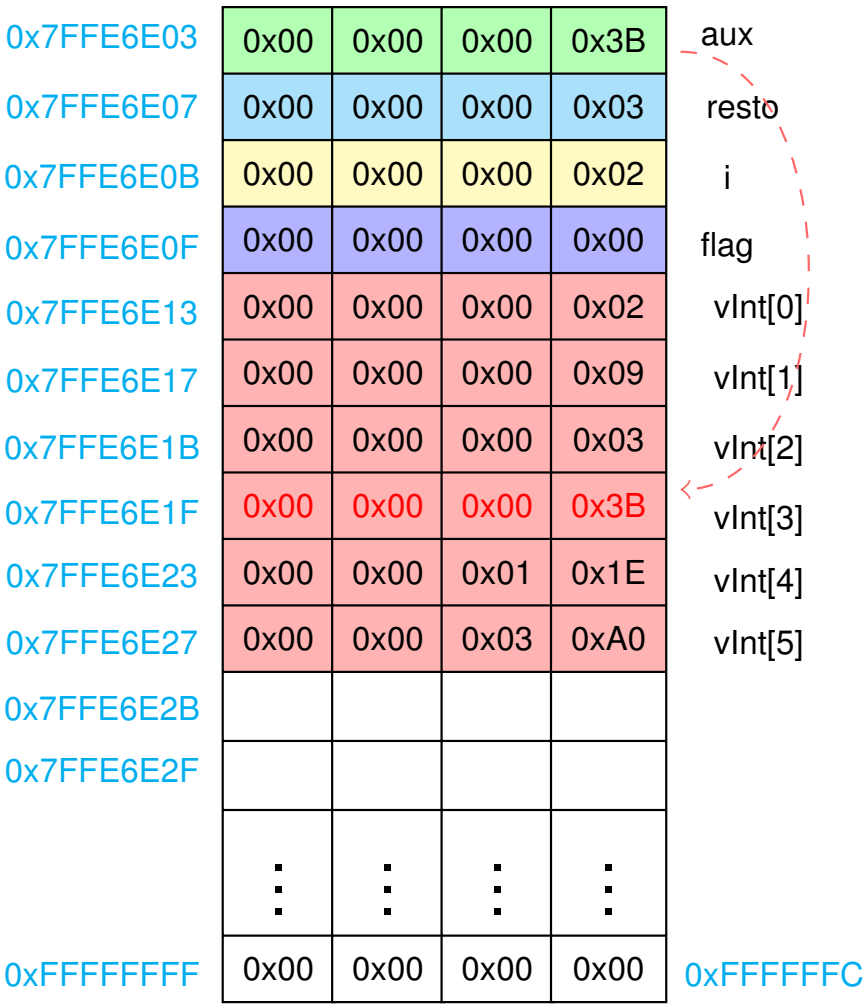



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

vInt[2] = vInt[3]

MEMORIA en ARQUITECTURA x86 32-bits

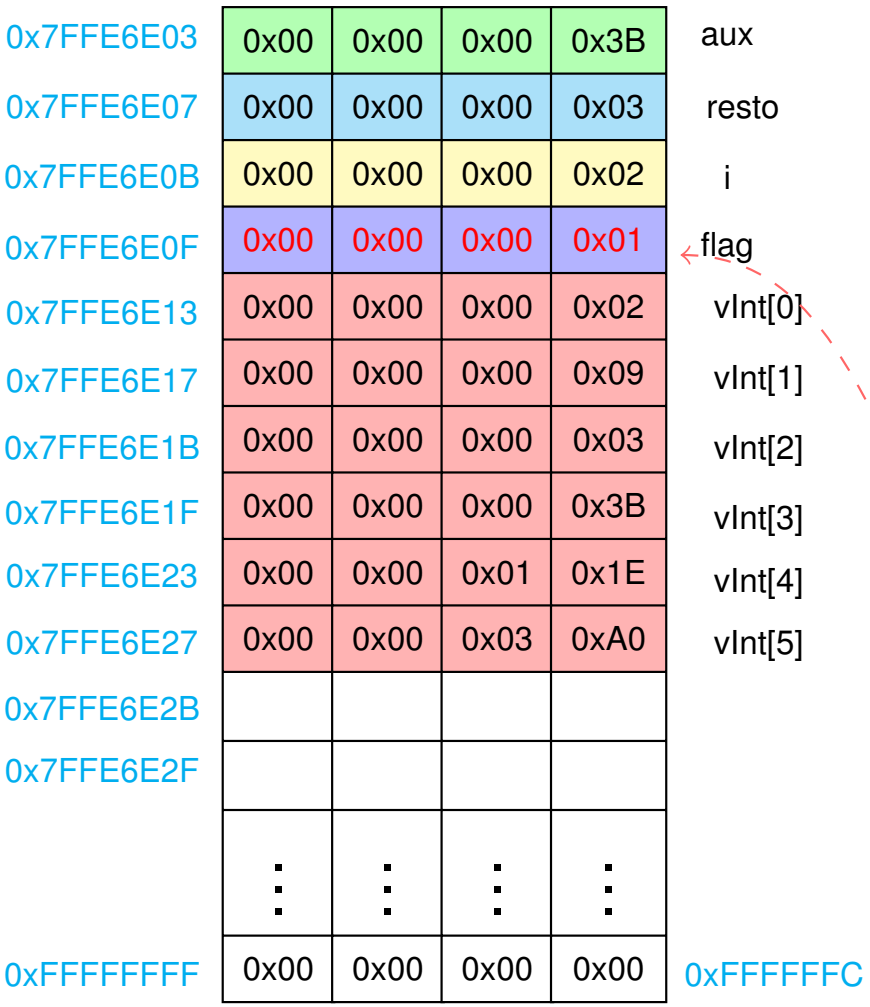



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

vInt[3] = aux

MEMORIA en ARQUITECTURA x86 32-bits

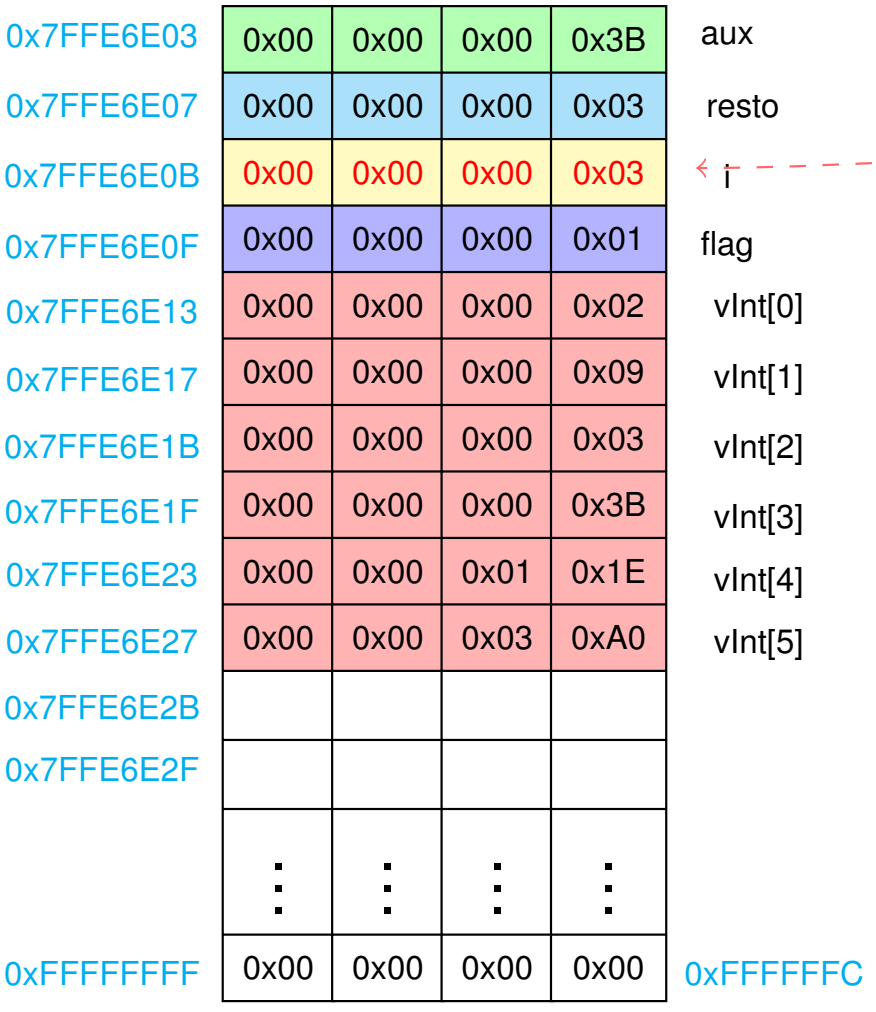



```c
#include <stdio.h>
#define CANT 6

int main (void)
{
int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
  resto=0;
  do
  {
    resto++;
    flag=0;
    for(i=0;i < (CANT-resto);i++)
    {
      if(vInt[i] > vInt[i+1])
      {
        aux=vInt[i];
        vInt[i]=vInt[i+1];
        vInt[i+1]=aux;
        flag=1;
      }
    }
  }while(flag);
return (0);
}
```

6-3=3

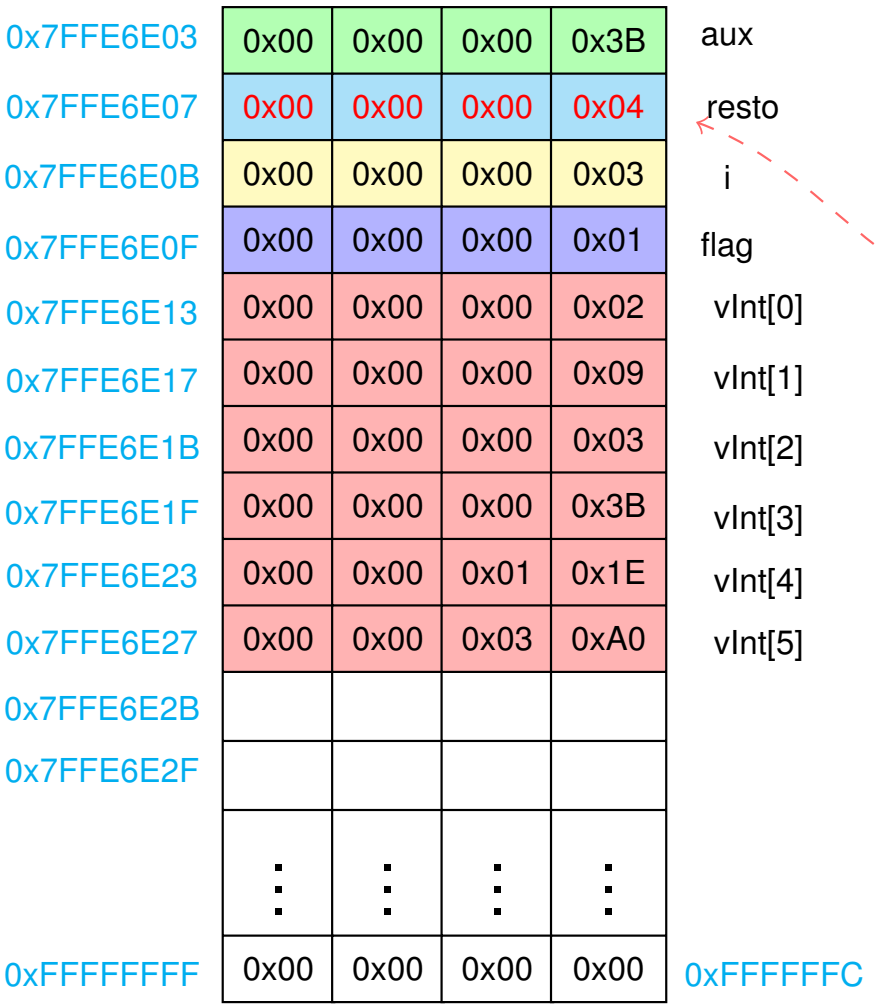
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

MEMORIA en ARQUITECTURA x86 32-bits

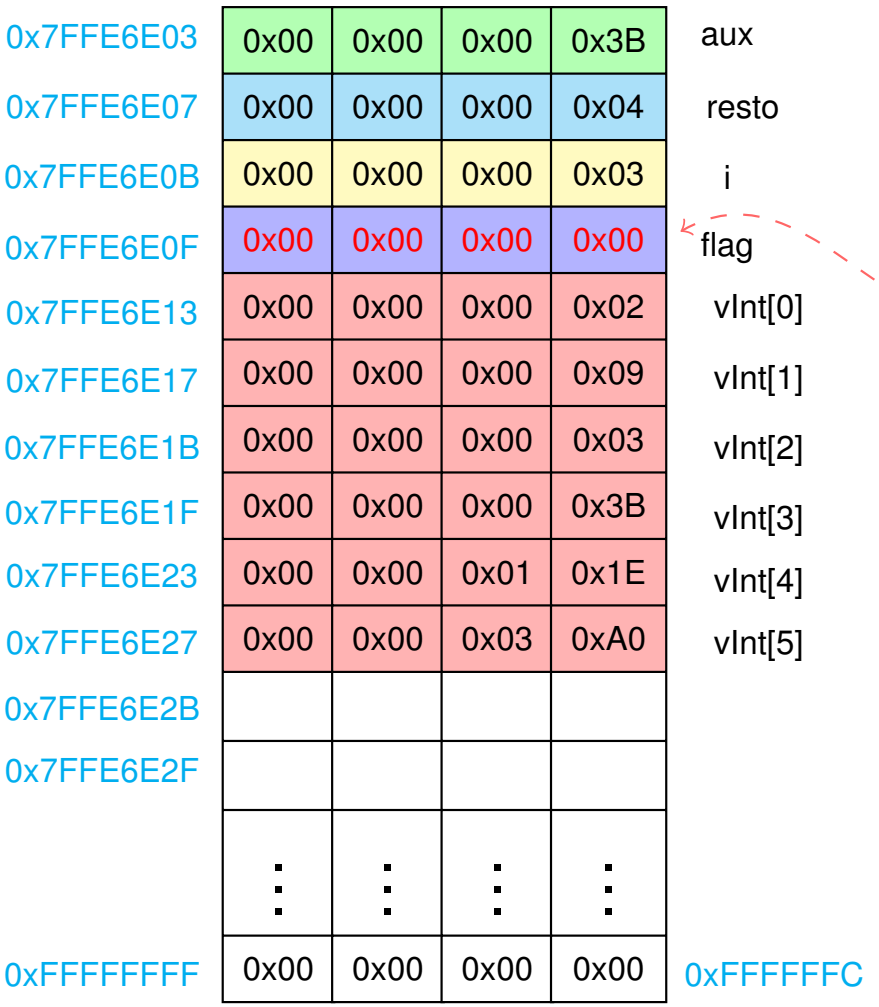



```c
#include <stdio.h>
#define CANT 6

int main (void)
{
int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
  resto=0;
  do
  {
    resto++;
    flag=0;
    for(i=0;i < (CANT-resto);i++)
    {
      if(vInt[i] > vInt[i+1])
      {
        aux=vInt[i];
        vInt[i]=vInt[i+1];
        vInt[i+1]=aux;
        flag=1;
      }
    }
  }while(flag);
return (0);
}
```

6-3=3

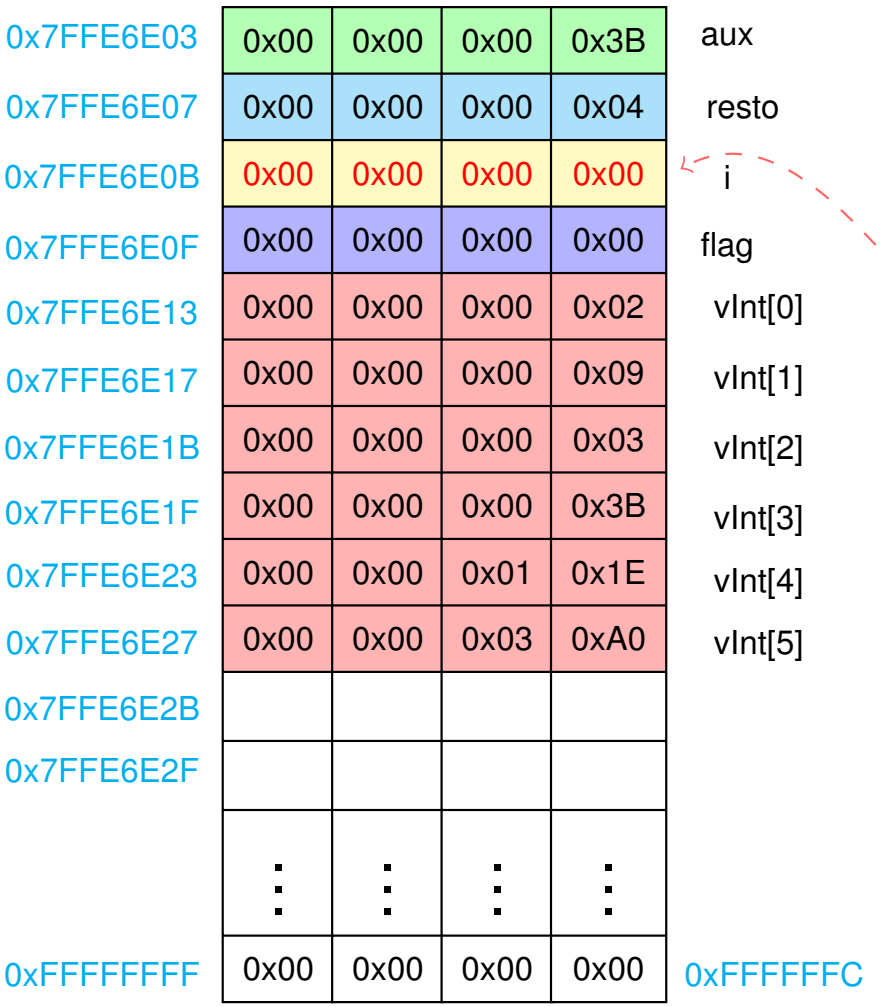
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-3=3

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

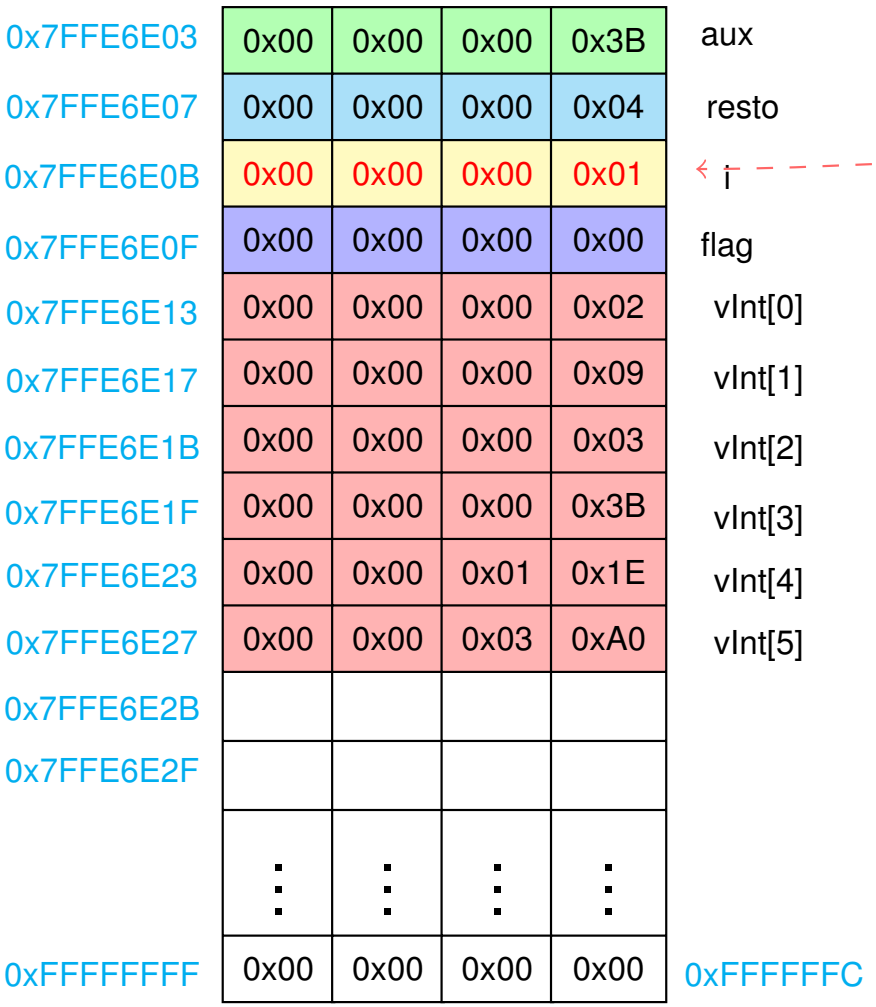
MEMORIA en ARQUITECTURA x86 32-bits


```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

0 1

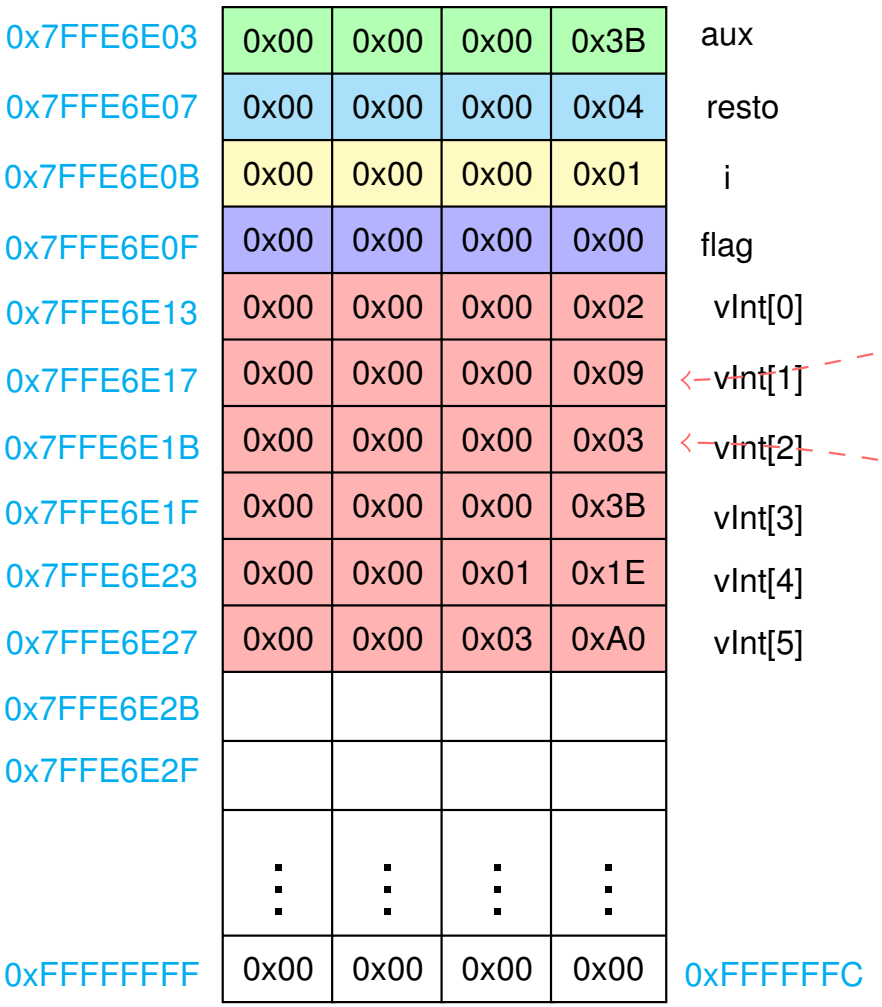
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

MEMORIA en ARQUITECTURA x86 32-bits

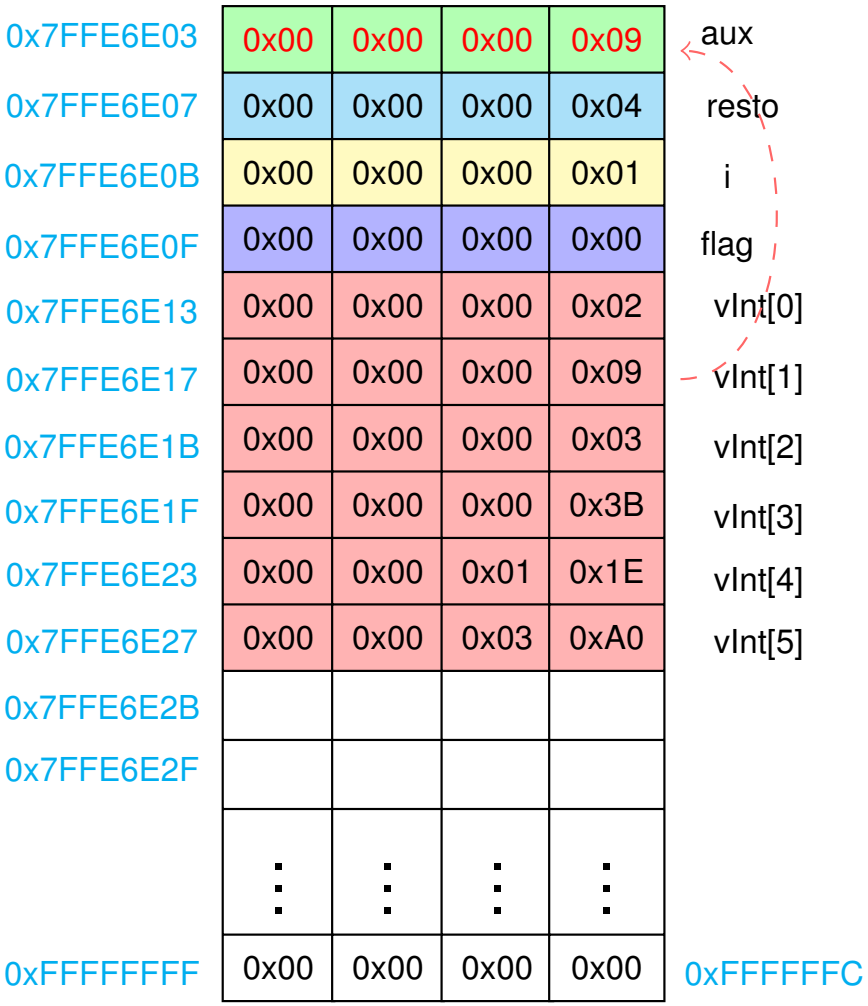



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

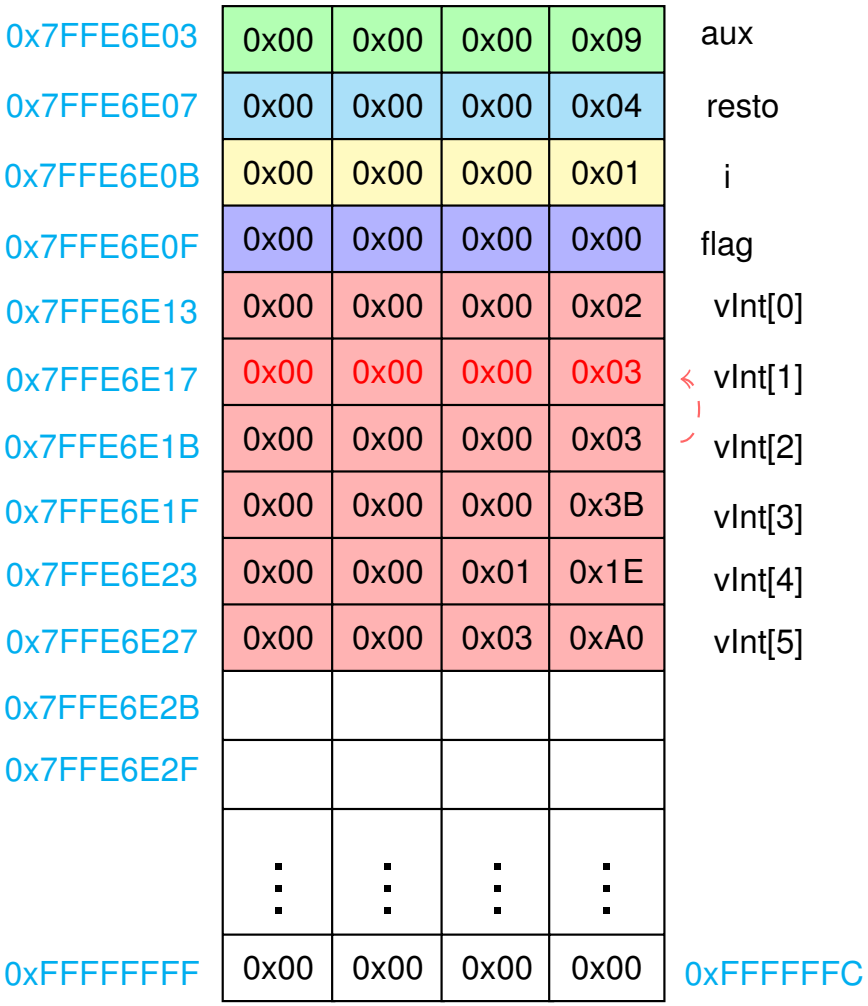
1 2

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-4=2
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {              1
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

MEMORIA en ARQUITECTURA x86 32-bits

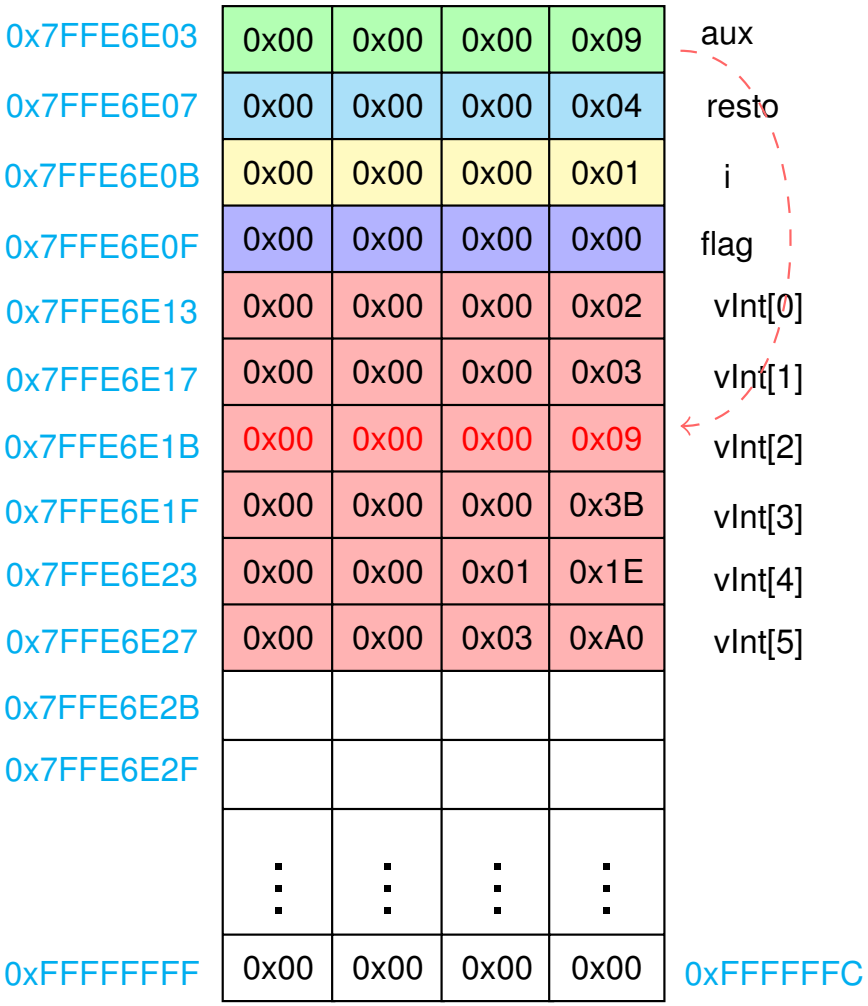



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

vInt[1] = vInt[2]

MEMORIA en ARQUITECTURA x86 32-bits

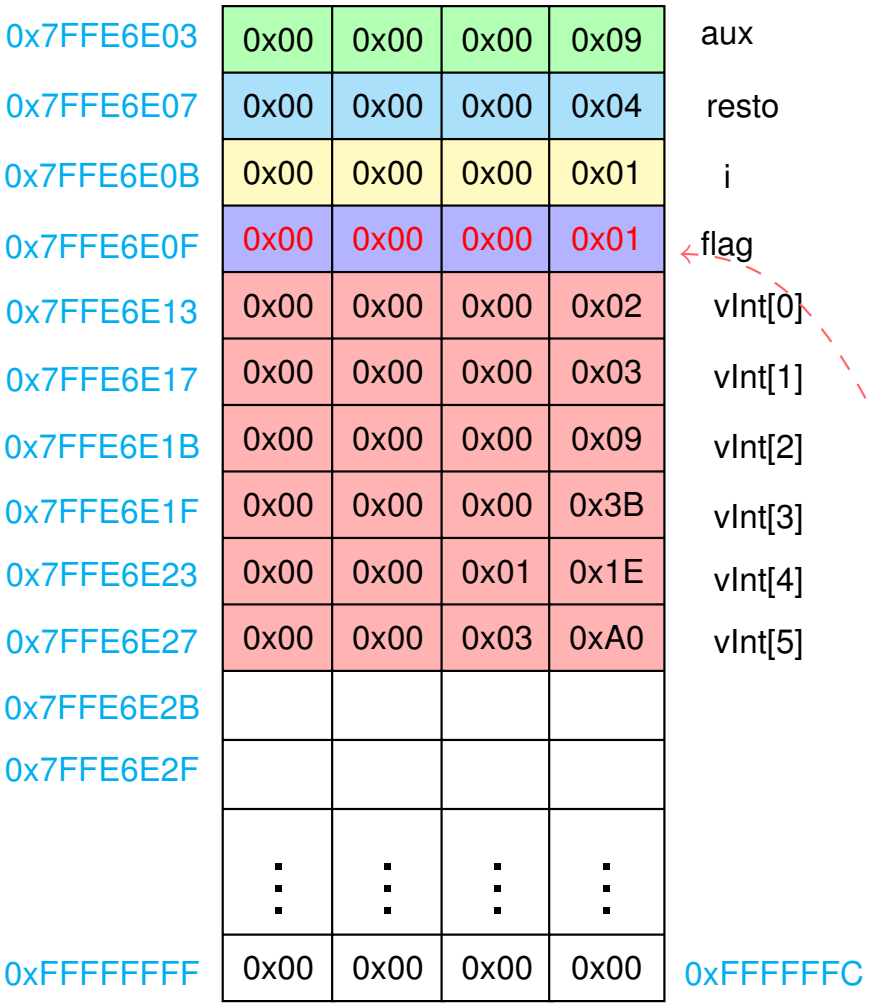



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

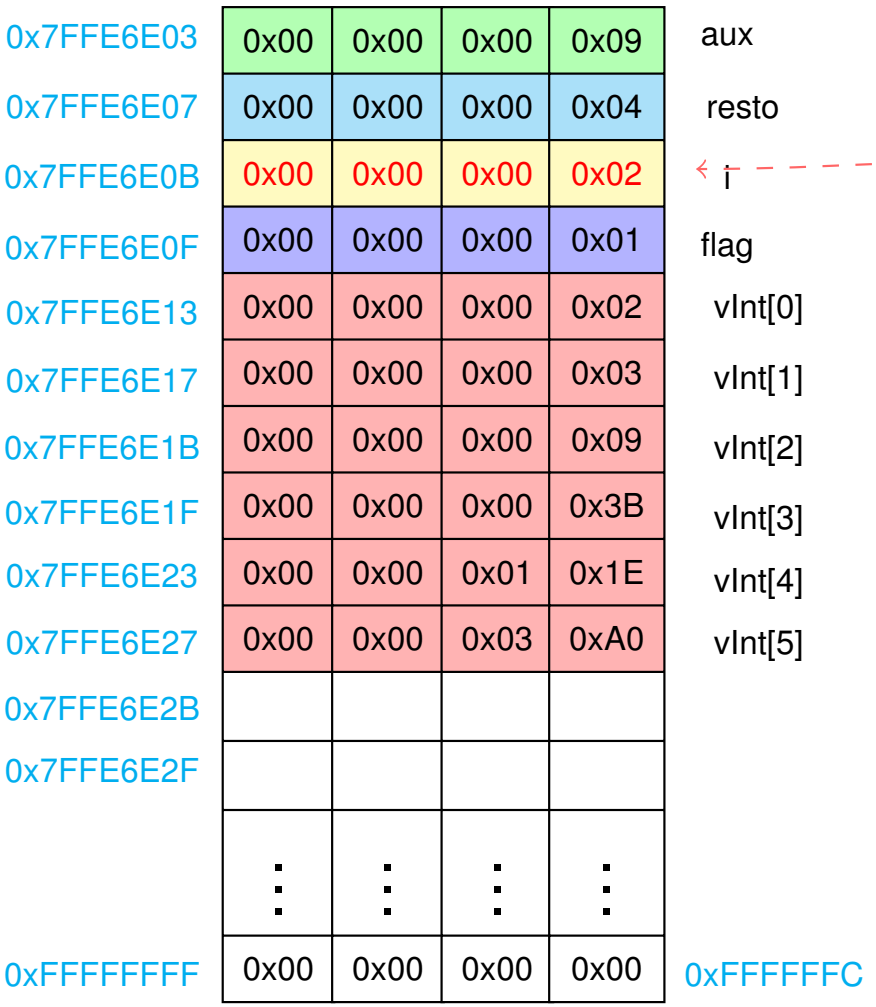
vInt[2] = aux

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;                    6-4=2
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

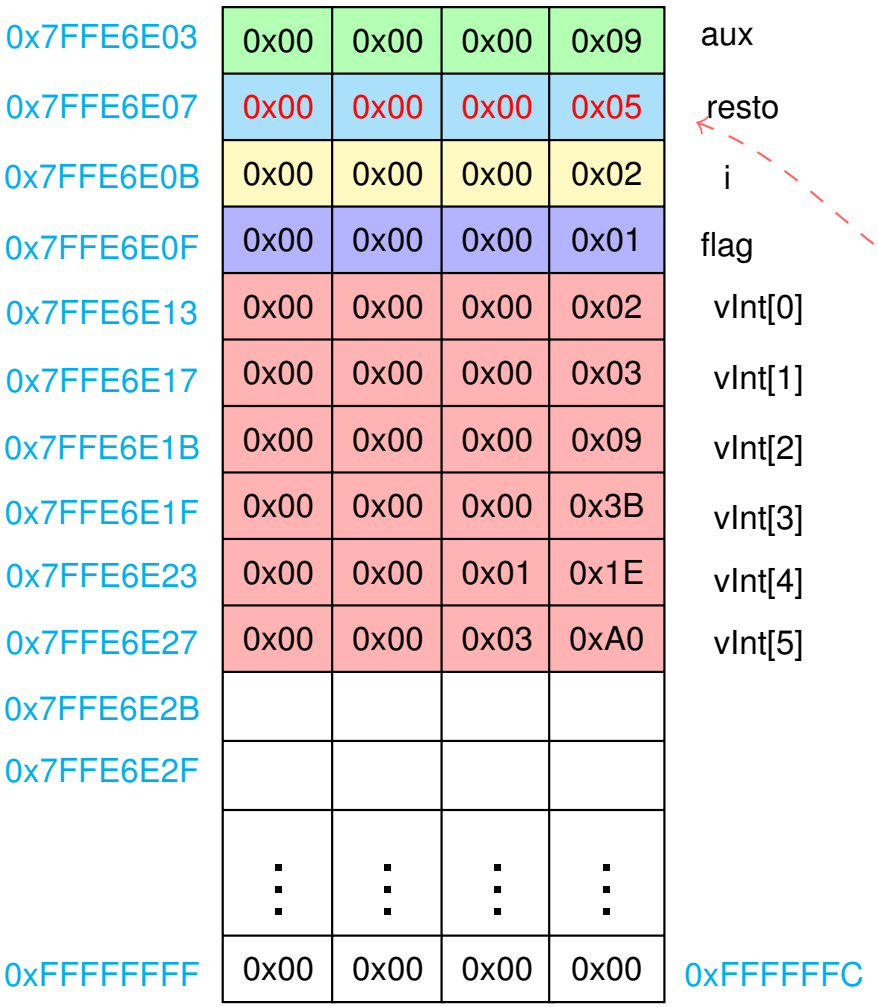
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

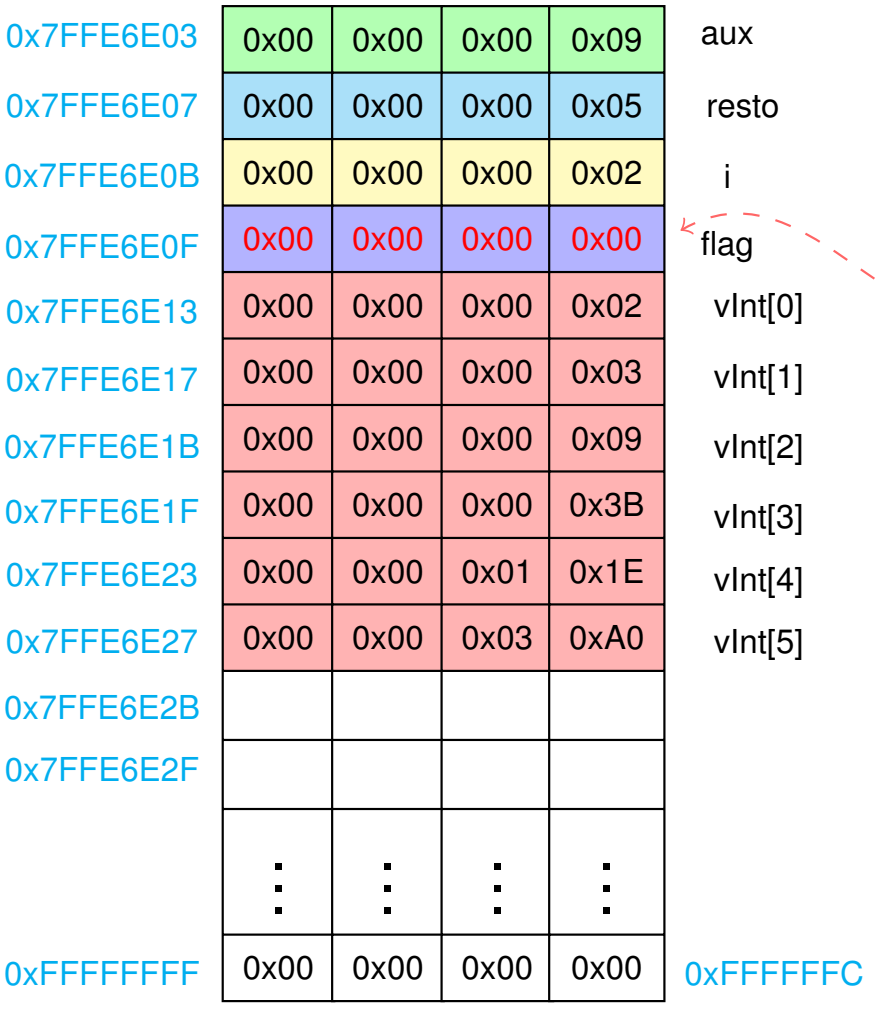
6-4=2

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;          6-4=2
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

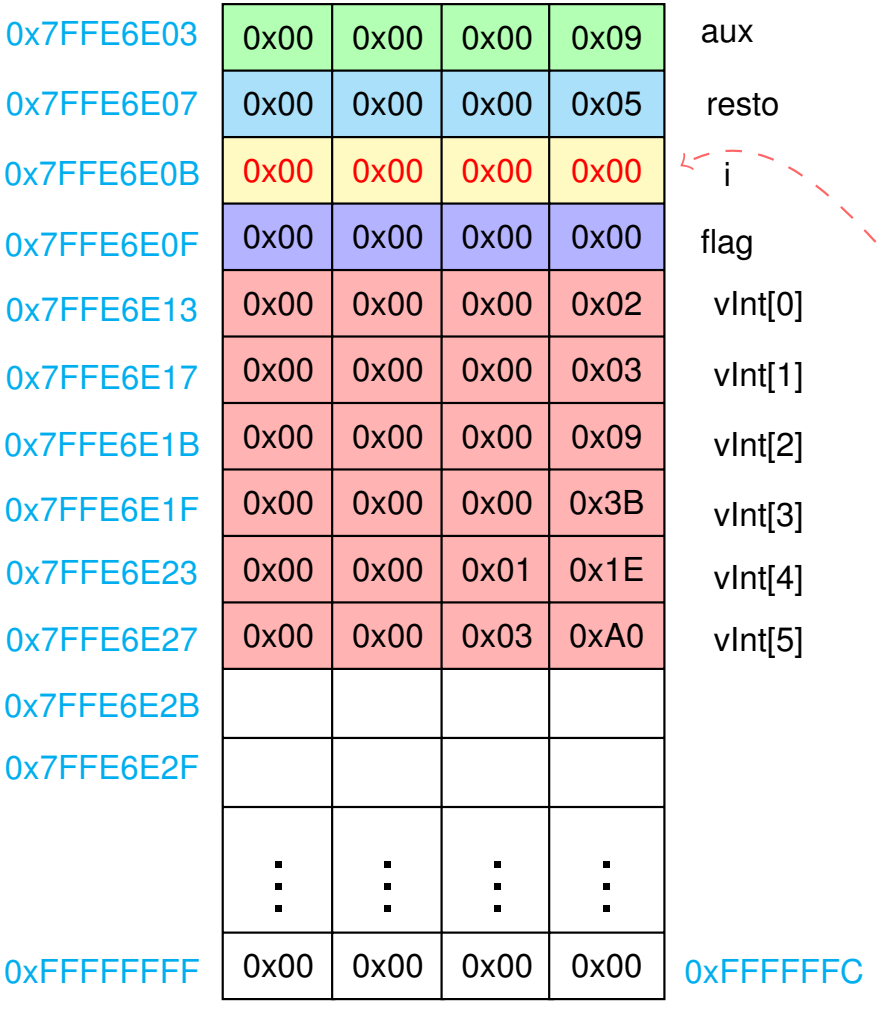
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-4=2

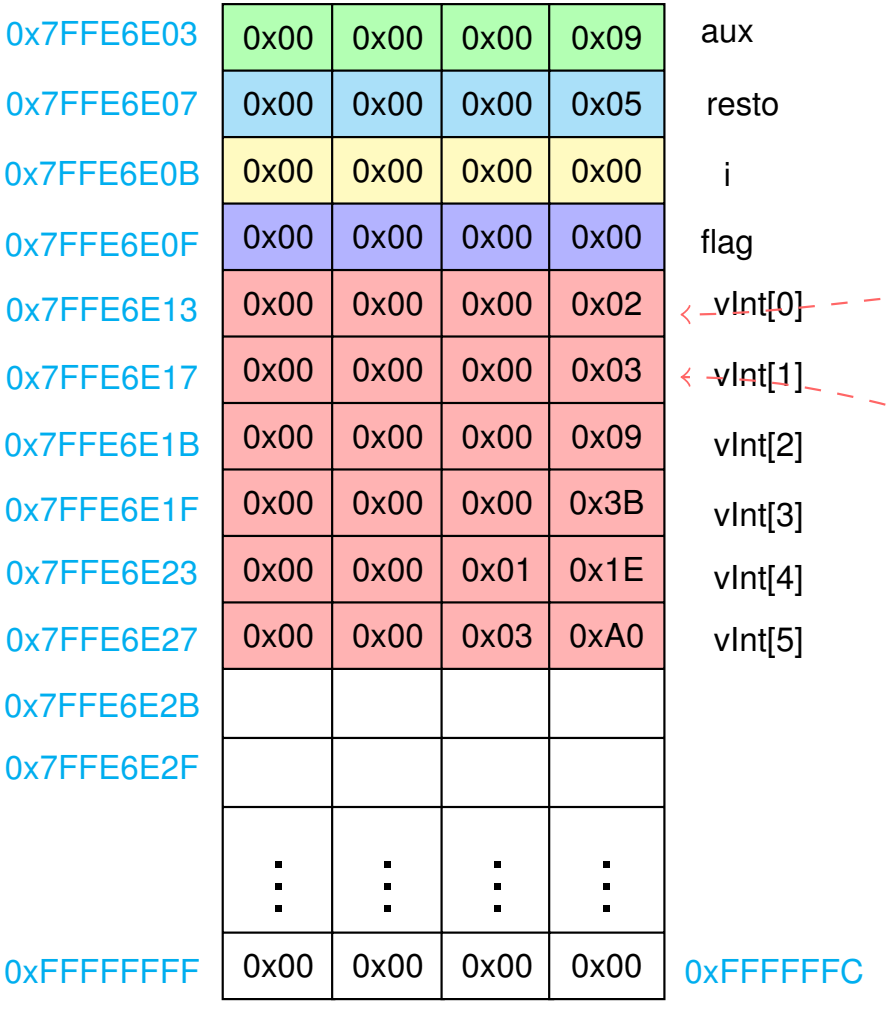
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-5=1

MEMORIA en ARQUITECTURA x86 32-bits

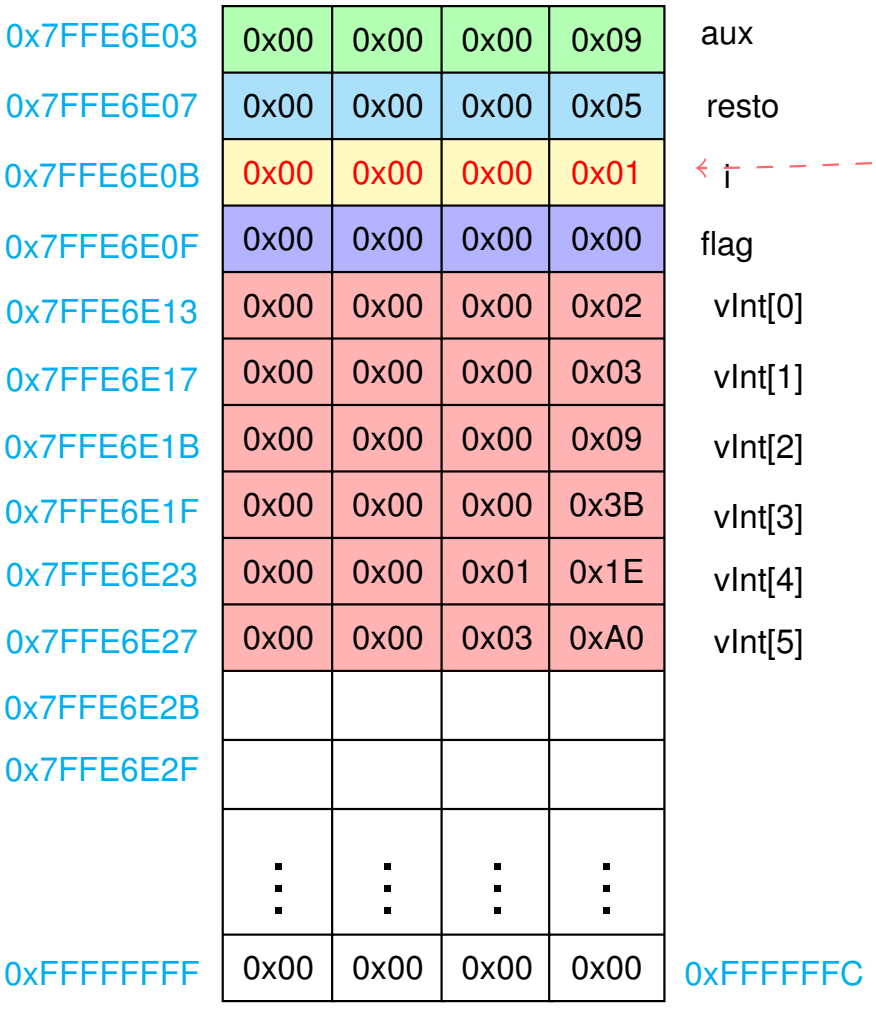



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-5=1

0 1

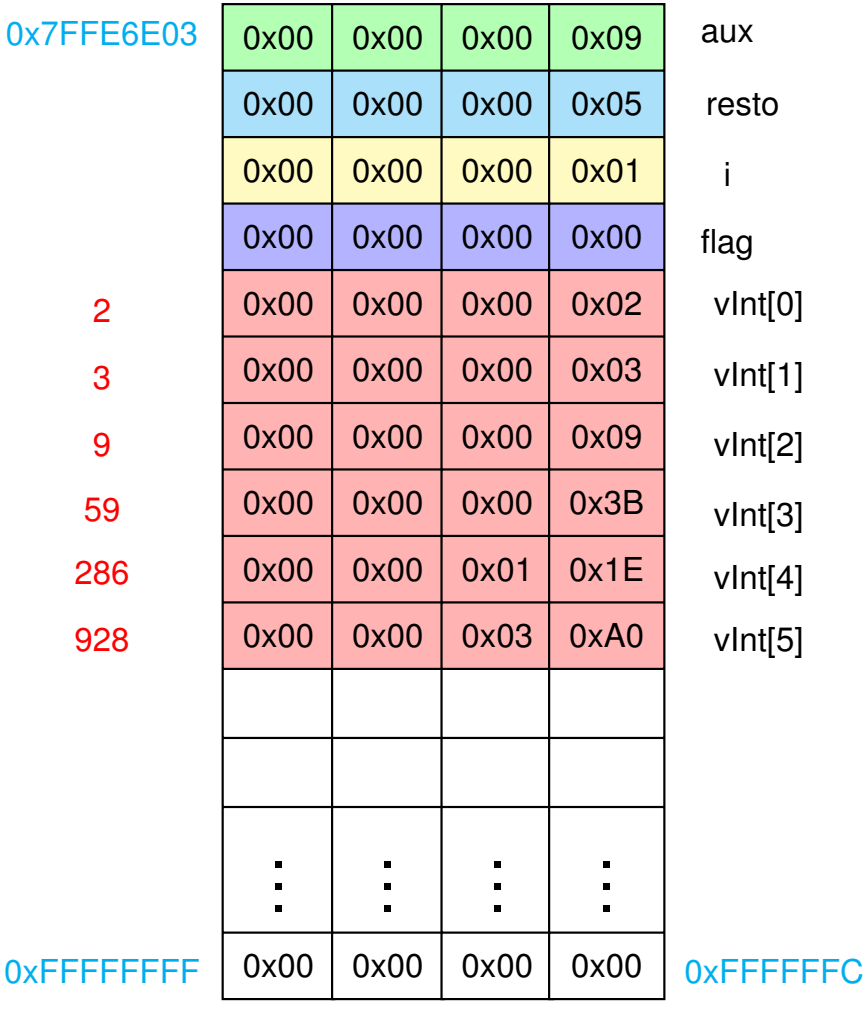
MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

6-5=1

MEMORIA en ARQUITECTURA x86 32-bits

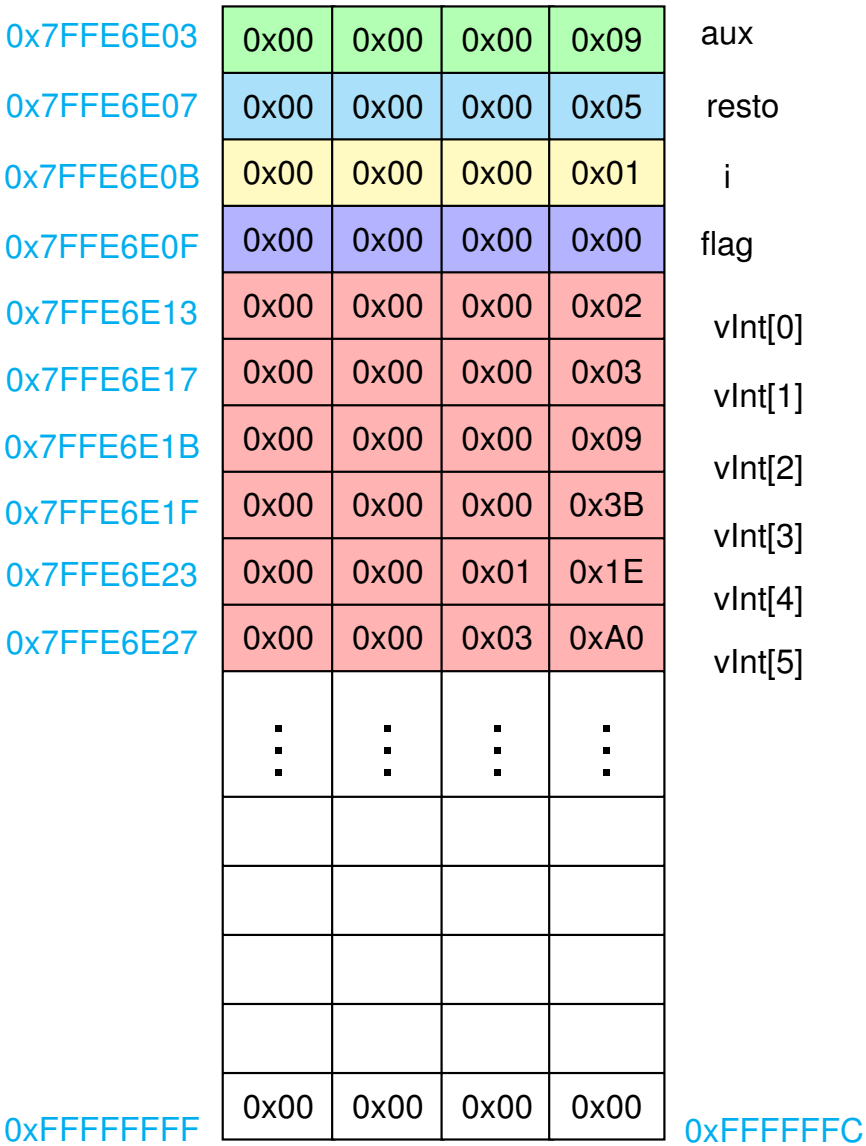



```
1  #include <stdio.h>
2  #define CANT 6
3
4  int main (void)
5  {
6  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
7    resto=0;
8    do
9    {
10     resto++;
11     flag=0;
12     for(i=0;i < (CANT-resto);i++)
13     {
14       if(vInt[i] > vInt[i+1])
15       {
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23 return (0);
24 }
```

Con while(0) sale del loop do-while y termina

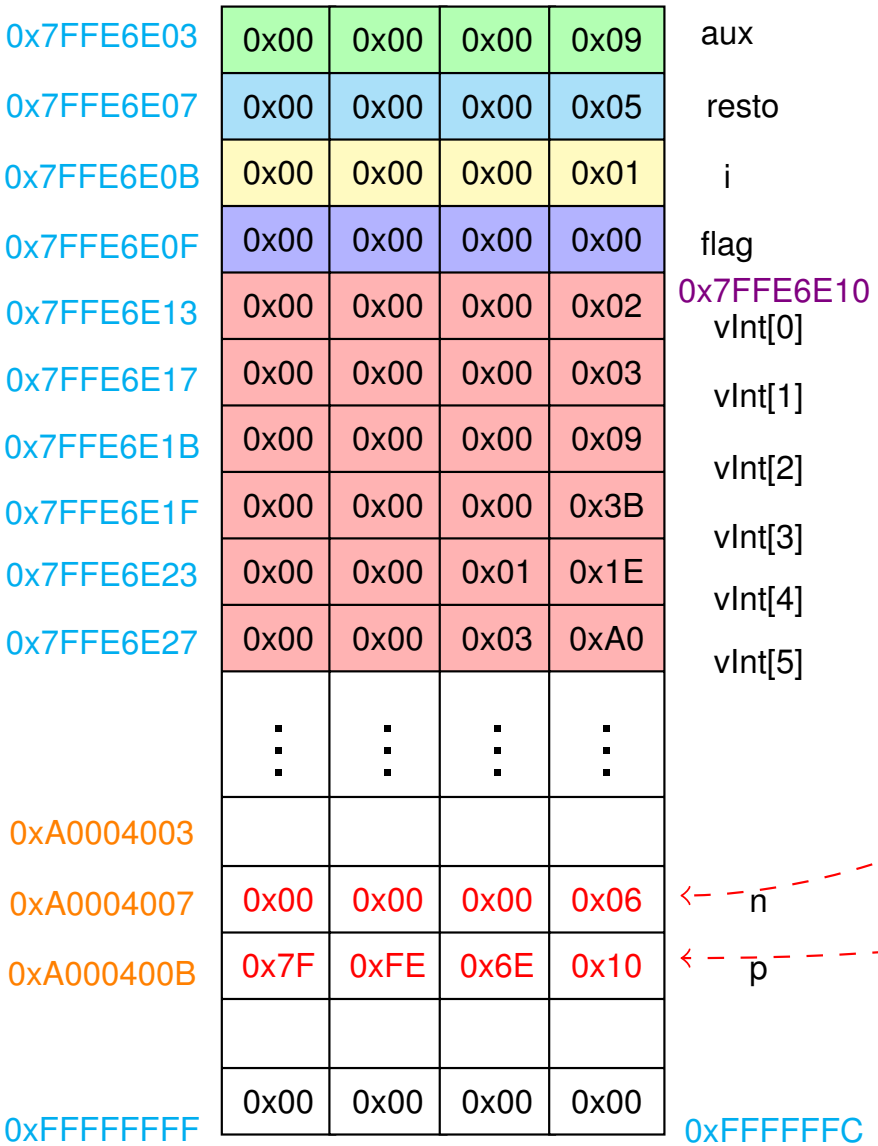
Verificar valores con la función *imprimir*

MEMORIA en ARQUITECTURA x86 32-bits



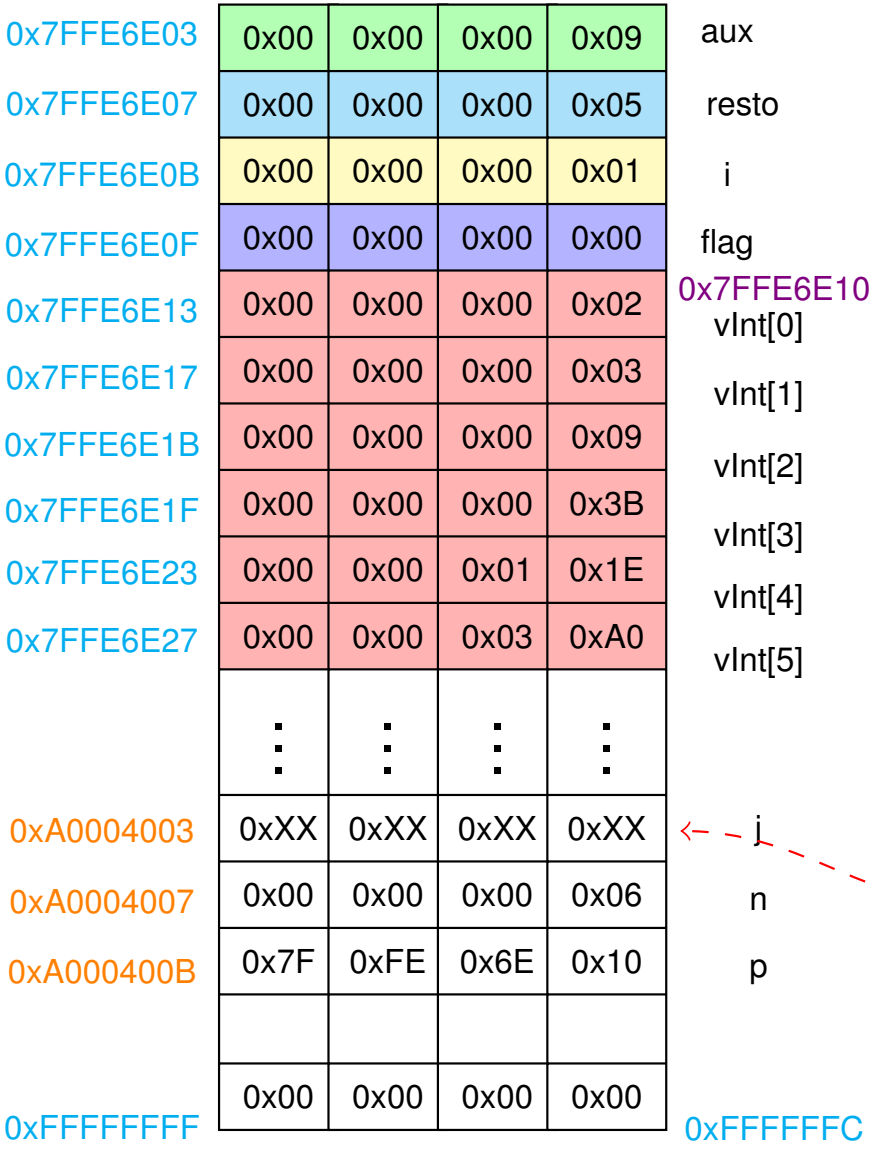

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



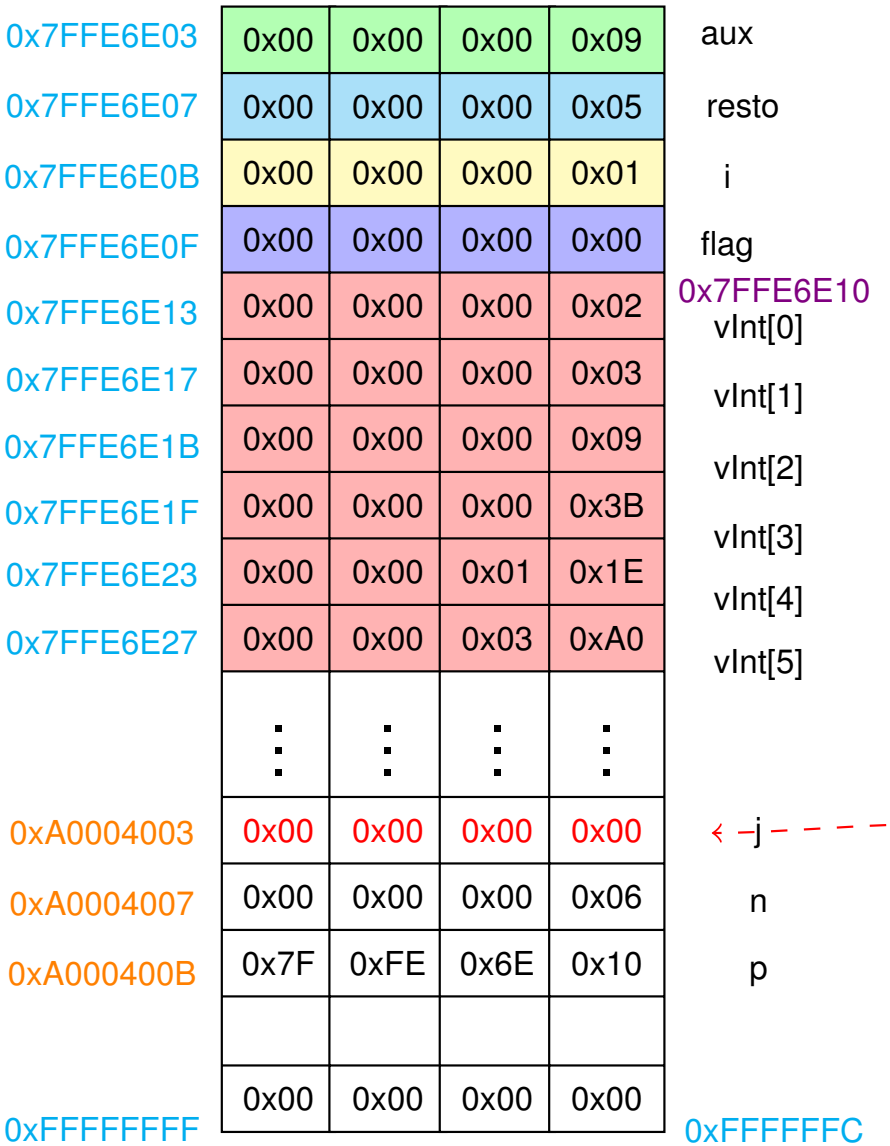

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30     printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



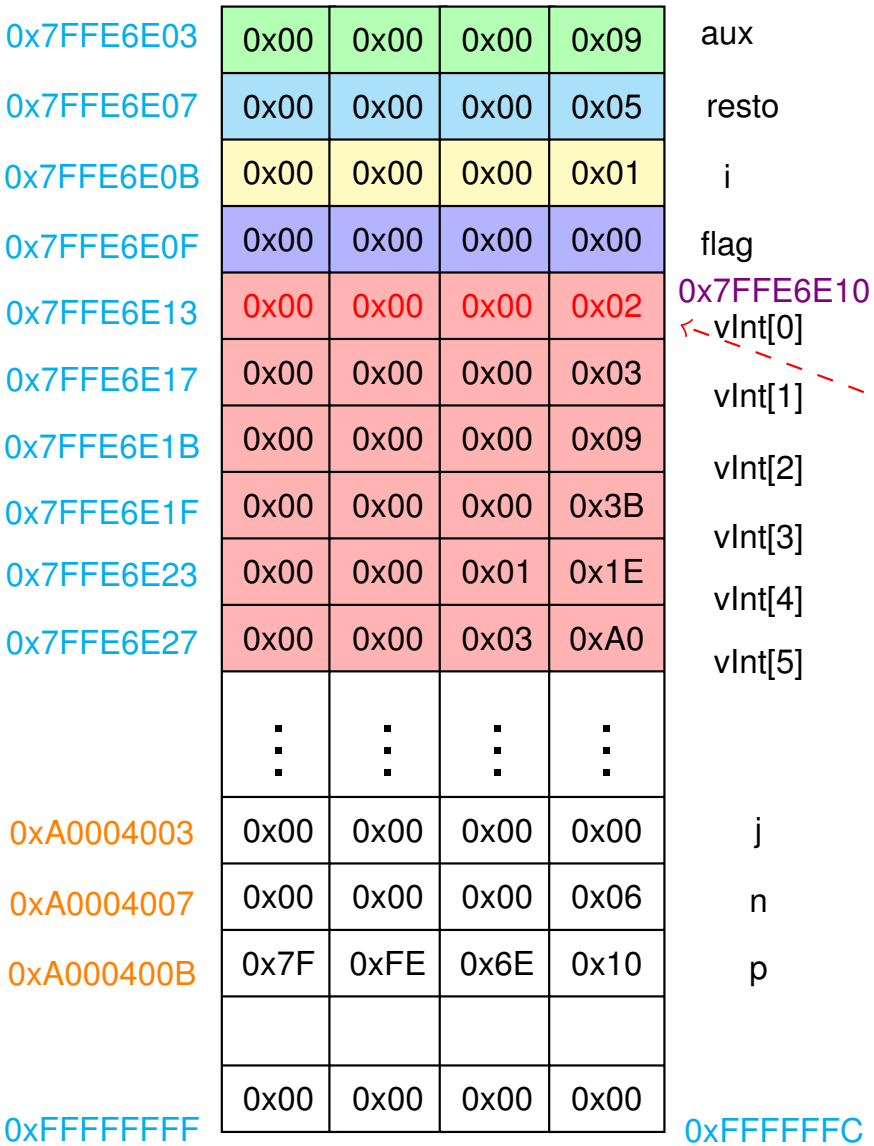

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



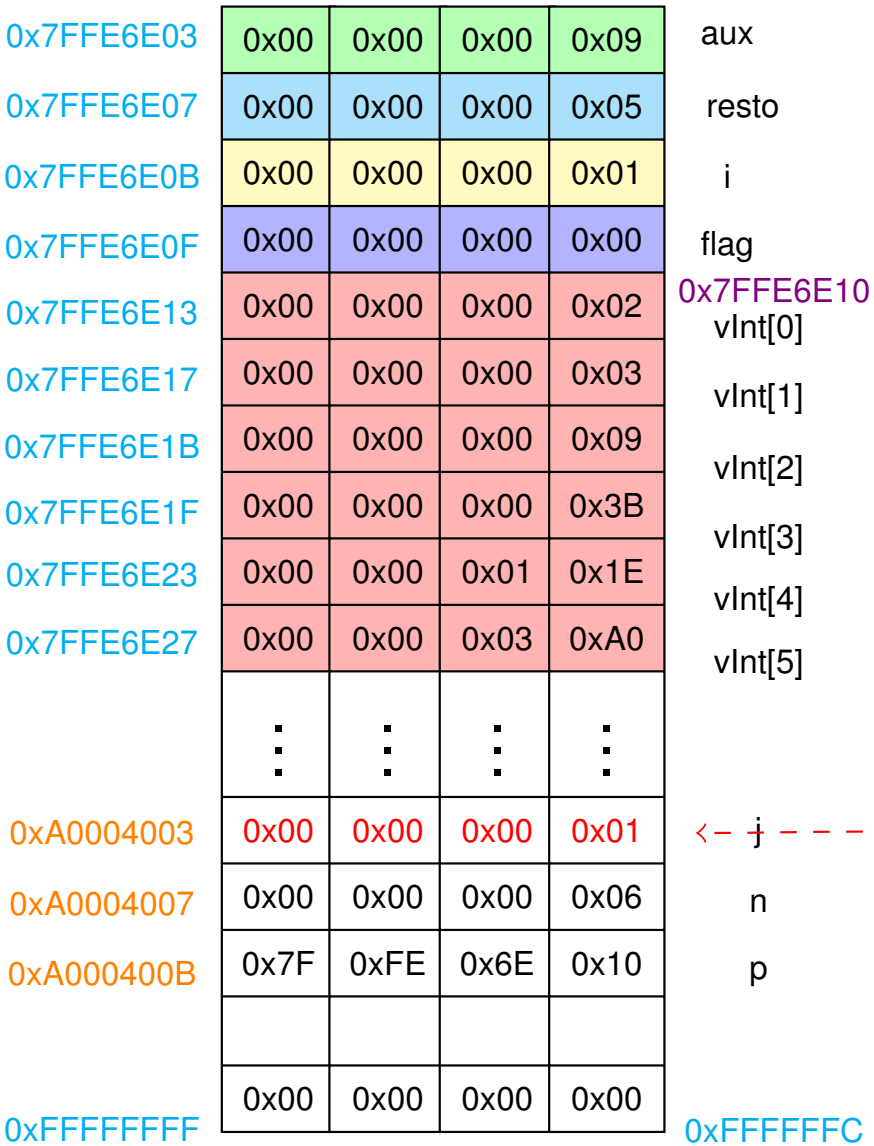

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



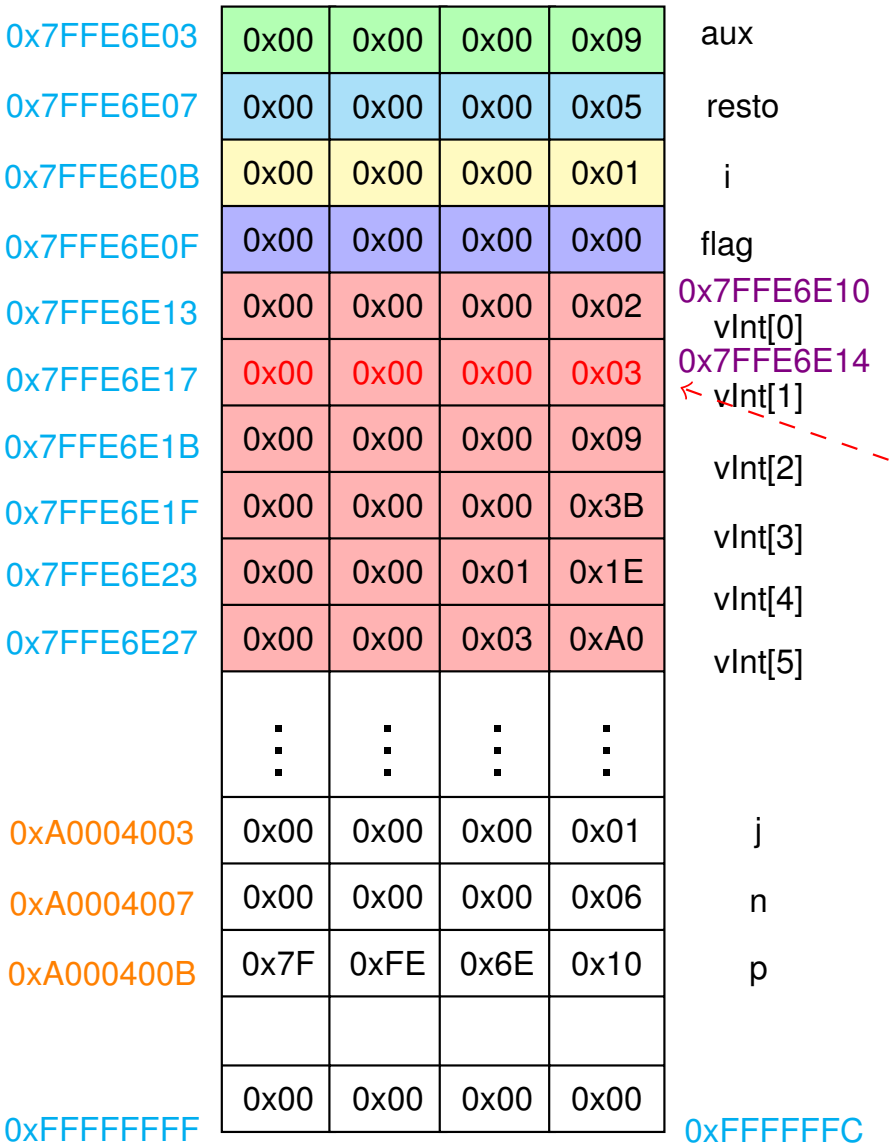

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



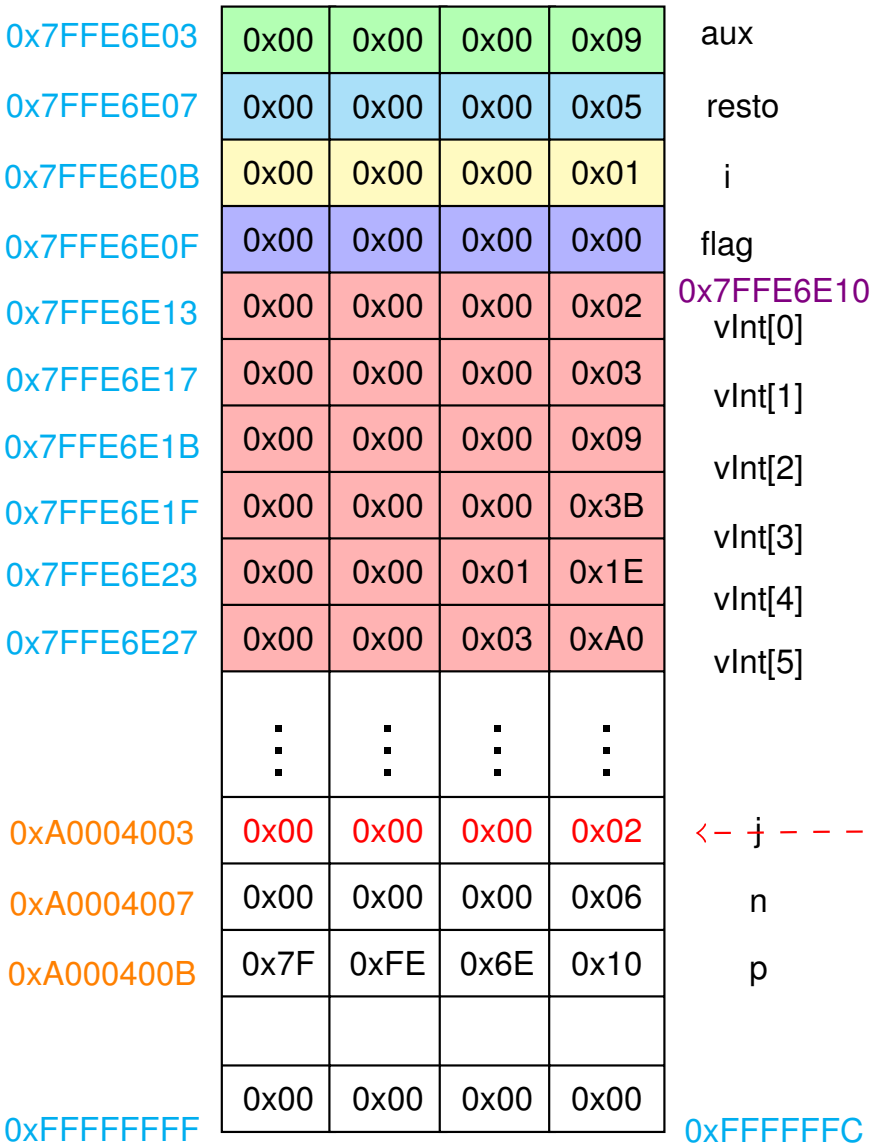

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



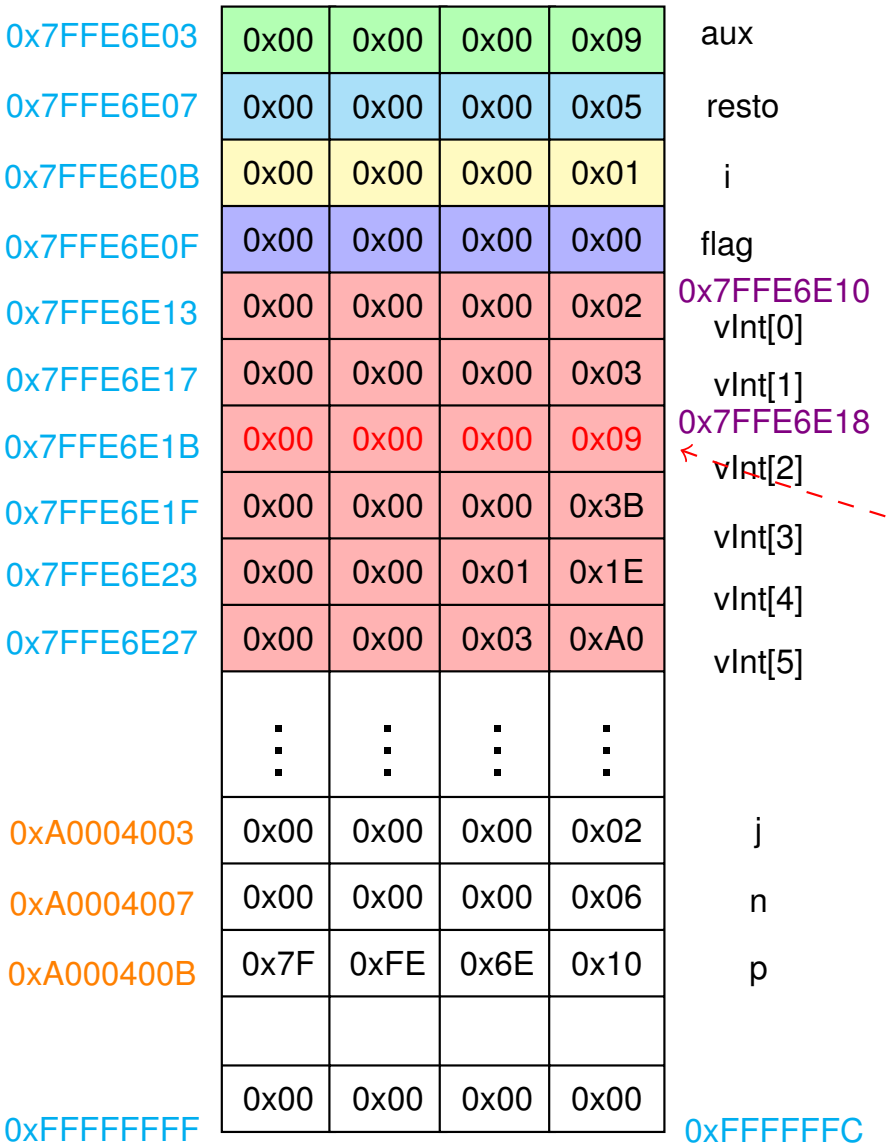

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



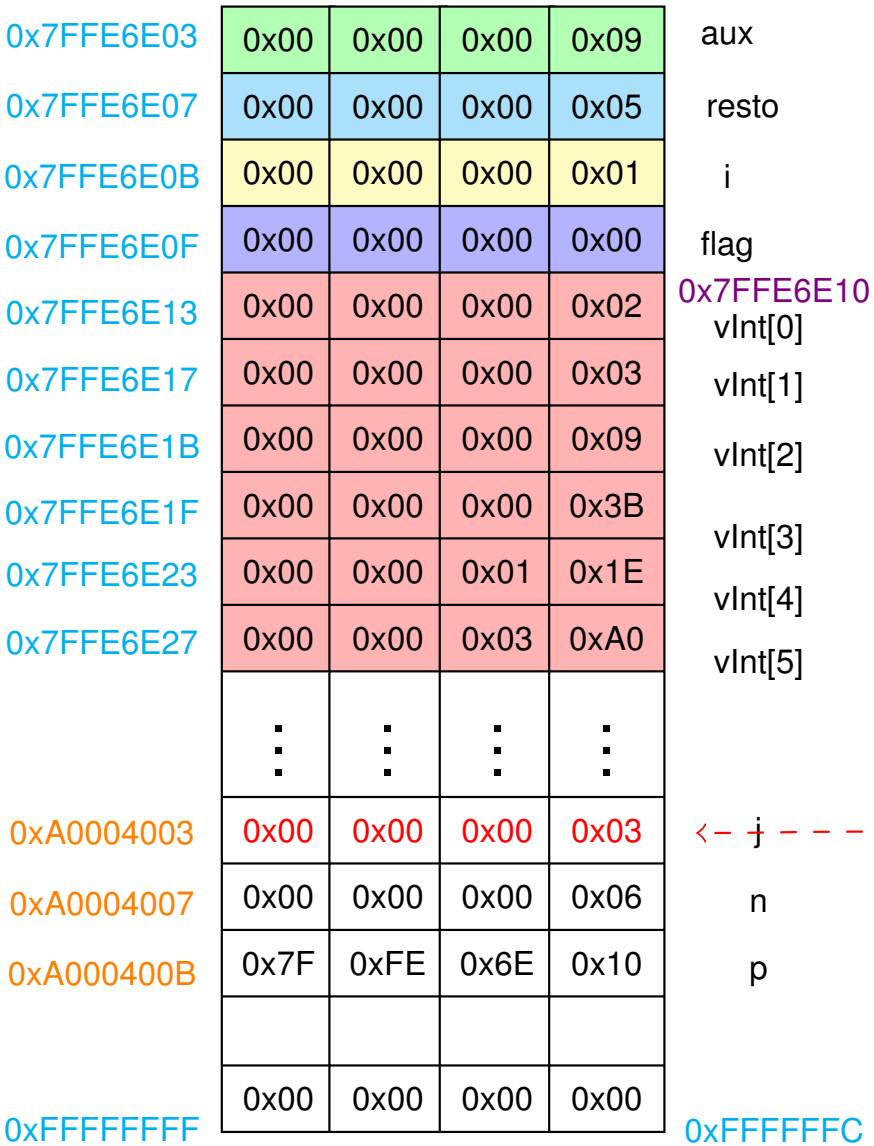

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



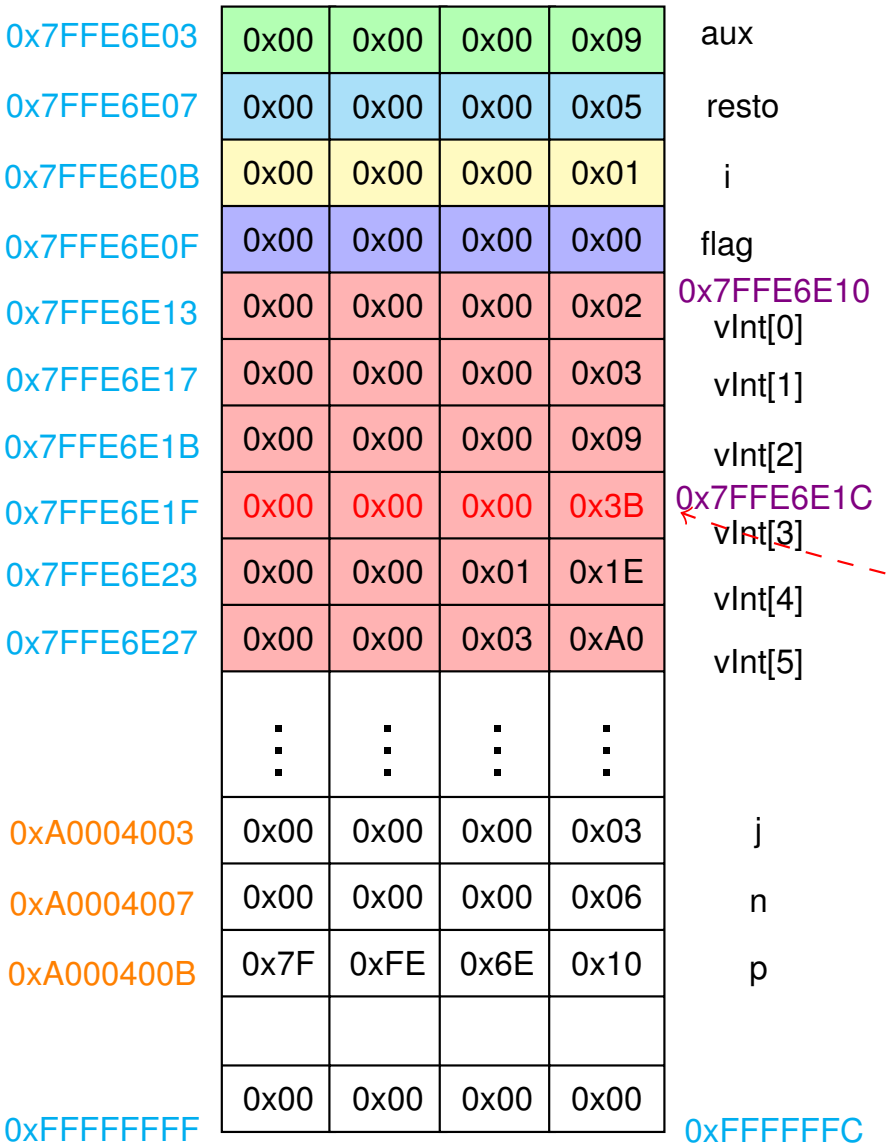

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



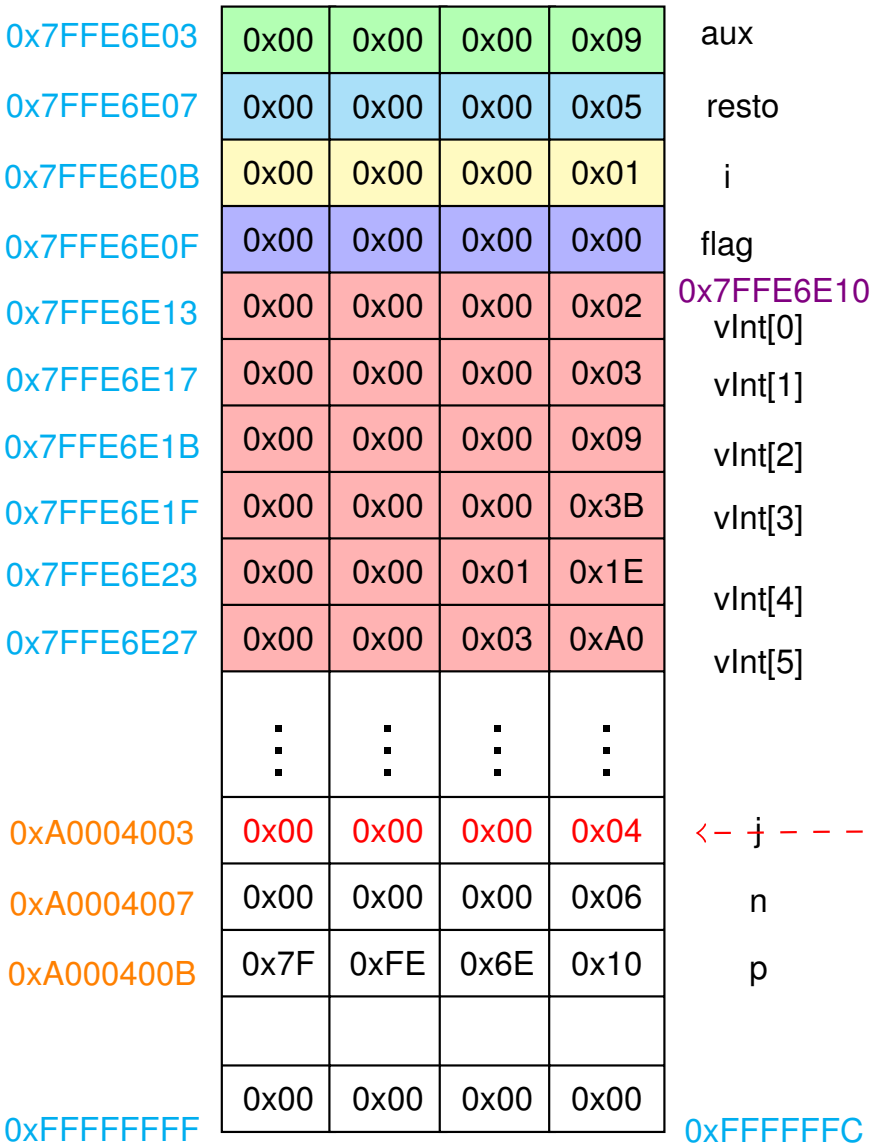

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



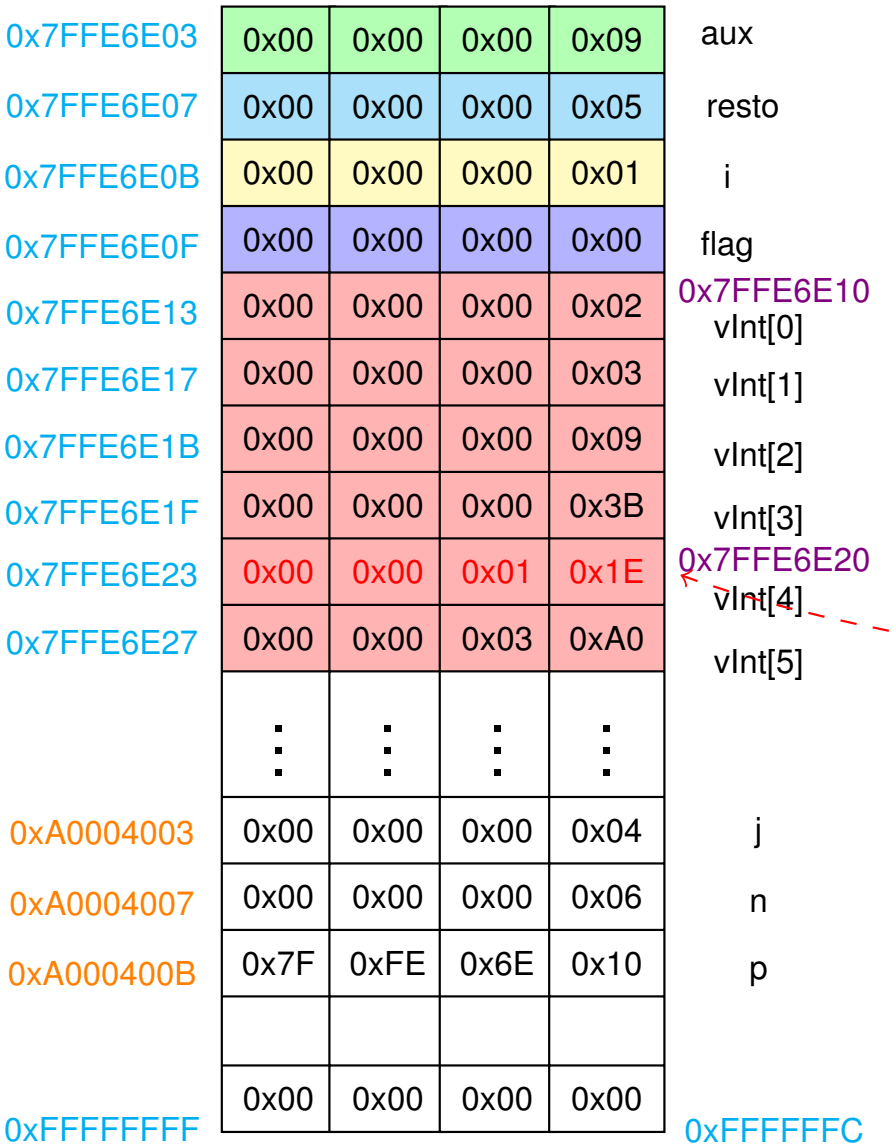

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



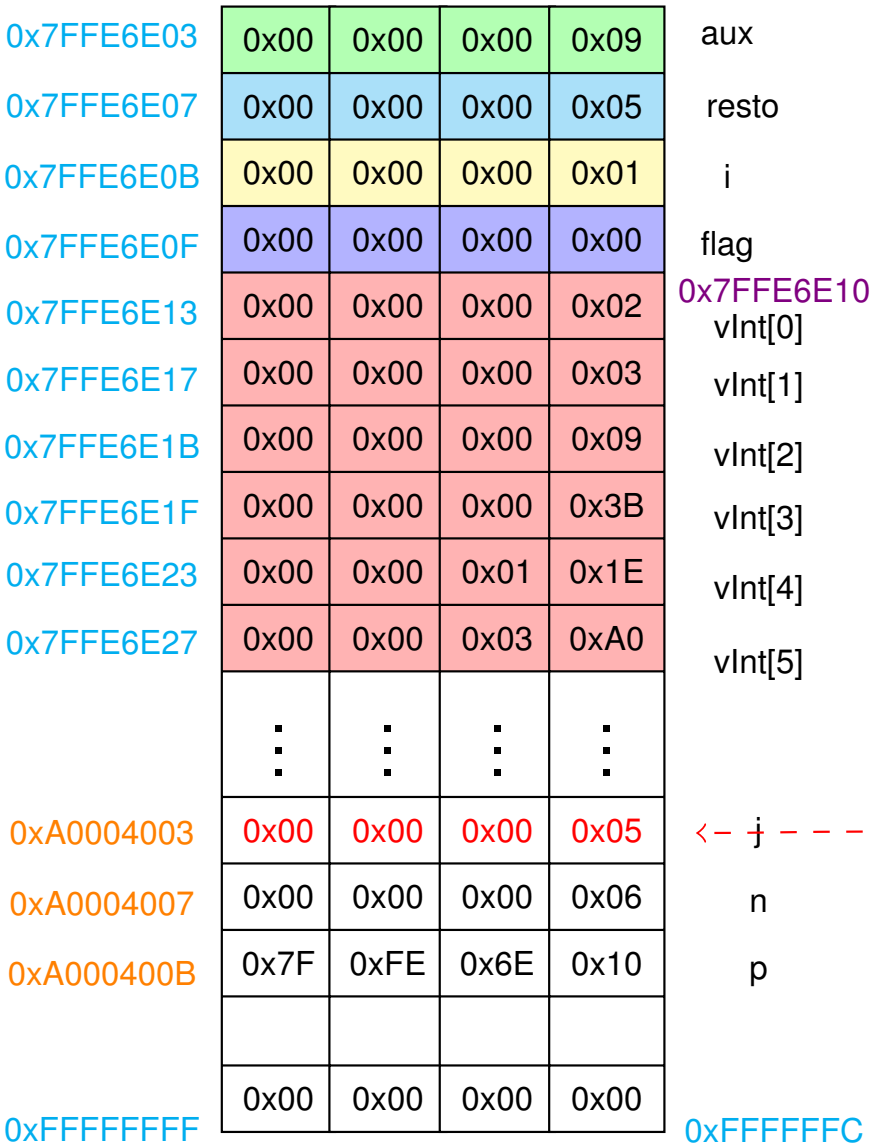

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



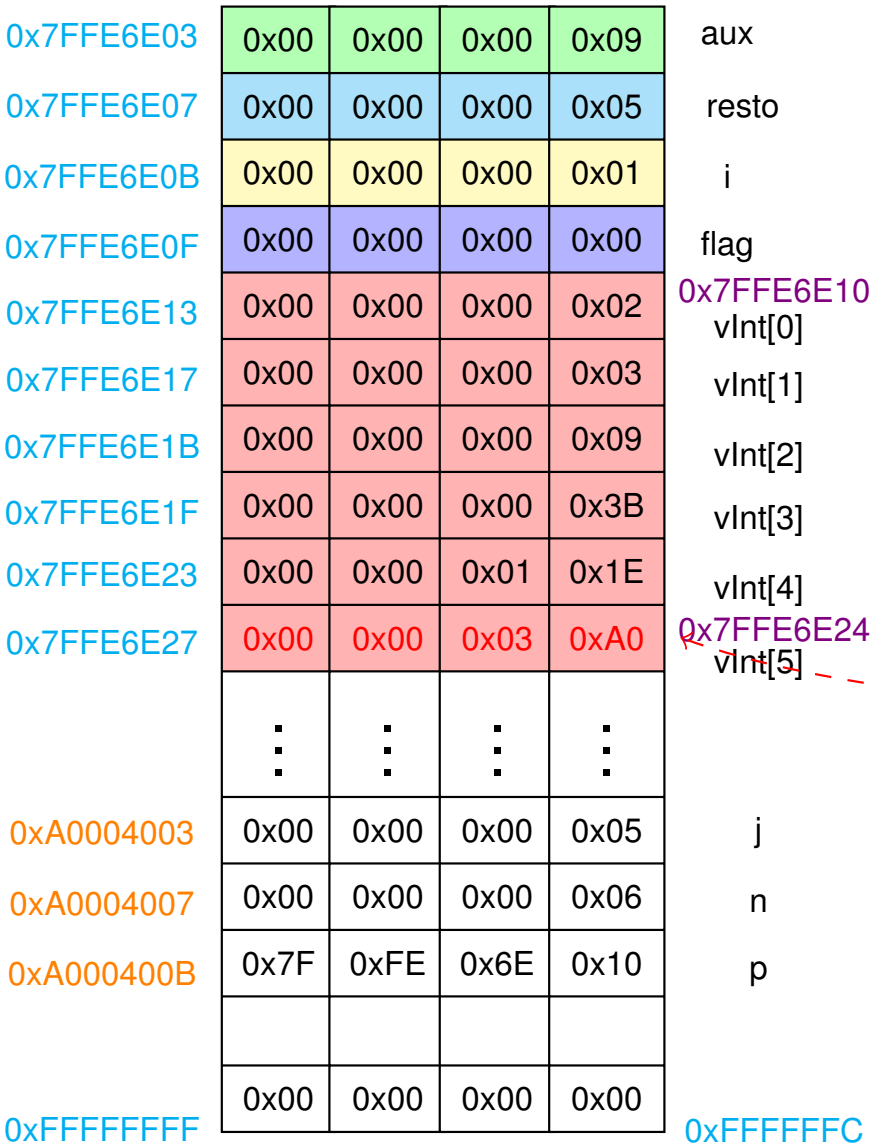

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits



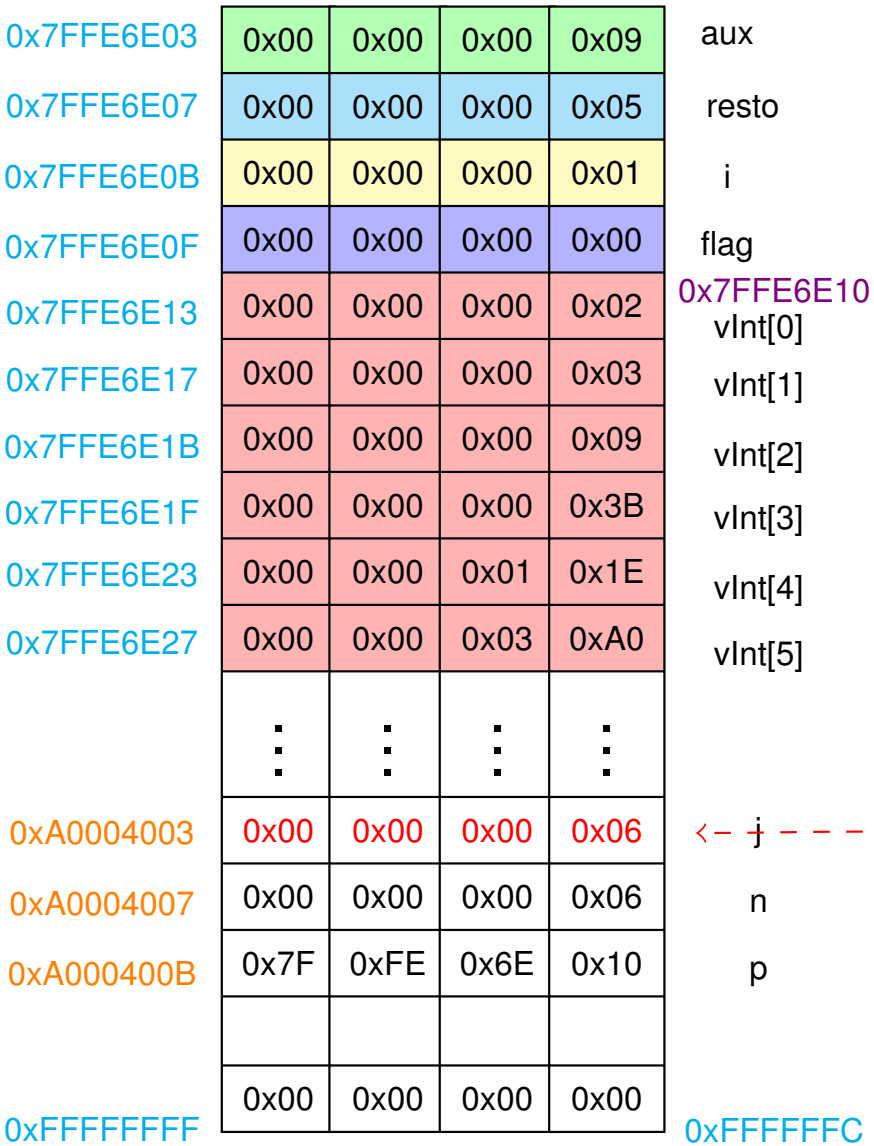

```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits

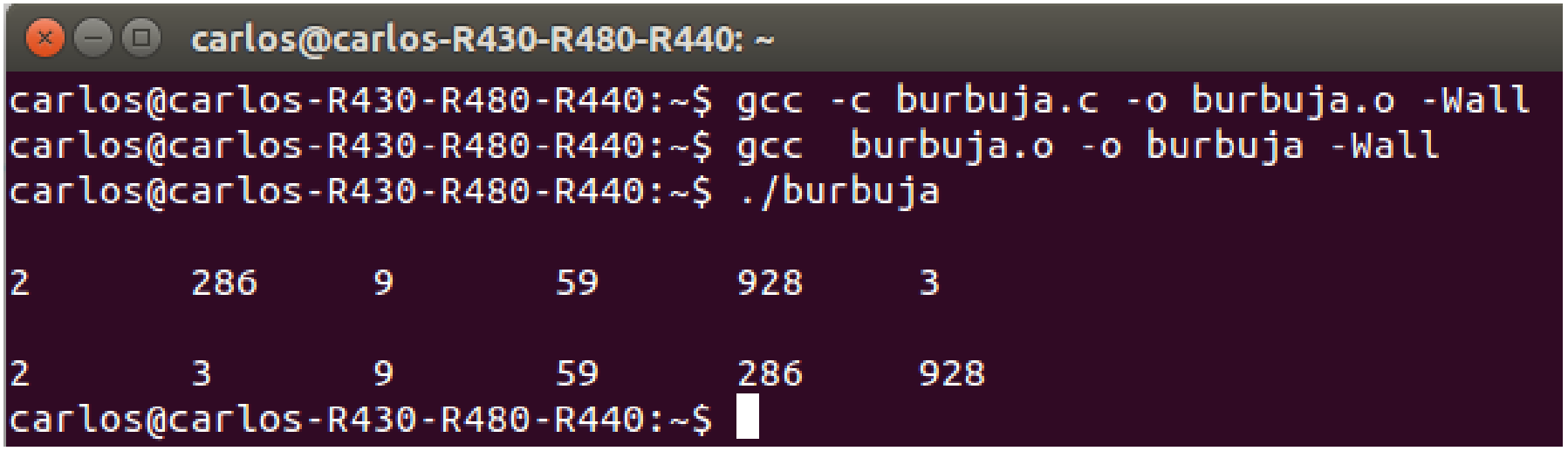



```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```

MEMORIA en ARQUITECTURA x86 32-bits




```
1  #include <stdio.h>
2  #define CANT 6
3
4  void imprimir(int *p,int n);
5
6  int main (void)
7  {
8  int aux,resto,i,flag,vInt[CANT]={2,286,9,59,928,3};
9    imprimir(&vInt[0],CANT);
10   resto=0;
11   do{
12     resto++;
13     flag=0;
14     for(i=0;i < (CANT-resto);i++){
15       if(vInt[i] > vInt[i+1]){
16         aux=vInt[i];
17         vInt[i]=vInt[i+1];
18         vInt[i+1]=aux;
19         flag=1;
20       }
21     }
22   }while(flag);
23   imprimir(&vInt[0],CANT);
24   return (0);
25 }
26 void imprimir( int*p,int n)
27 {
28   int j;
29   for(j=0;j<n;j++){
30    printf("%d \r\n",*(p+j));
31   }
32 }
```



- compila con ***gcc -c burbuja.c -o burbuja.o -Wall***
- Linkea con ***gcc burbuja.o -o burbuja -Wall***
- Ejecuta con ***./burbuja***
- Primero imprime en la consola los valores desordenados
- Al finalizar el ordenamiento también los imprime en la consola