Write the results of the following program execution:

```
#include <stdio.h>

int reset(void);
int next(int);
int last(int);
int new(int);

int i=1;

void main(void)
{
   auto int i,j=0;

   i=reset();
   /*   Local   i   kept equal to the same value.
      Function calls do not change it.
      And exactly that local  i  value is printed
      and passed to the functions (as their arguments
      value) during the consecutive calls.  */
   do {
      printf("i = %d\t",i),
      printf("j = %d\t",j), putchar('\n');
      printf("next(i) = %d\t",next(i)),
      putchar('\n');
      printf("last(i) = %d\t",last(i)),
      putchar('\n');
      printf("new(i+j) = %d\t",new(i+j)),
      putchar('\n');
      j++;
   } while (j<=2);
}

int reset(void)
{
   return i--;
   /*  Global   i   is decreased after
       setting the returning value of the
       reset   function, because  --
       operator is from the right-hand side
       of   i  */
}

int next(int j)
{
   return (j-=i--);
   /*   Global  i  is decreased after
      assignement since  --  operator
      is from the right-hand side of
      i  */
}

int last(int j)
{
   static int i=11;
   /*  Value of local  i  kept between
       the subsequent calls due to the
       static  declaration of  i  */
   return (i+=--j);
   /*  New local  i  set to value
       returned by function last.
       Decrementation of  j  takes place
       before assignement since it is from
       the left-hand side of  j  */
}

int new(int k)
{
   auto int j=4;
   return (i=j-=k);
   /*  New global  i  equal to value
       returned by function  new.
       To  i  is assigned the same value
       as to  j   */
}
```

Results:

```
i = 1 j = 0   (local  i  set to the global one
               before decrementation)
next(i) = 1
last(i) = 11
new(i+j) = 3
i = 1 j = 1
next(i) = -2
last(i) = 11
new(i+j) = 2
i = 1 j = 2
next(i) = -1
last(i) = 11
new(i+j) = 1
```