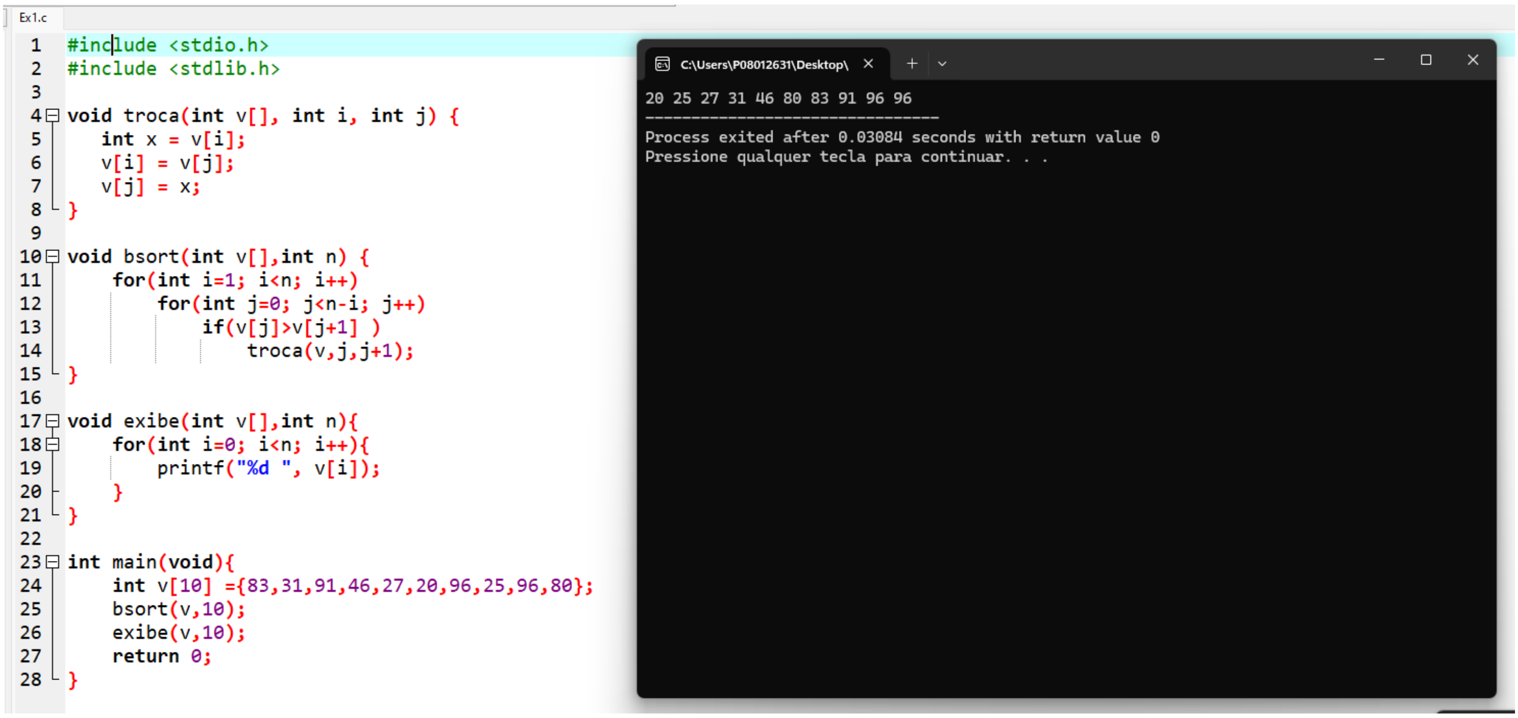
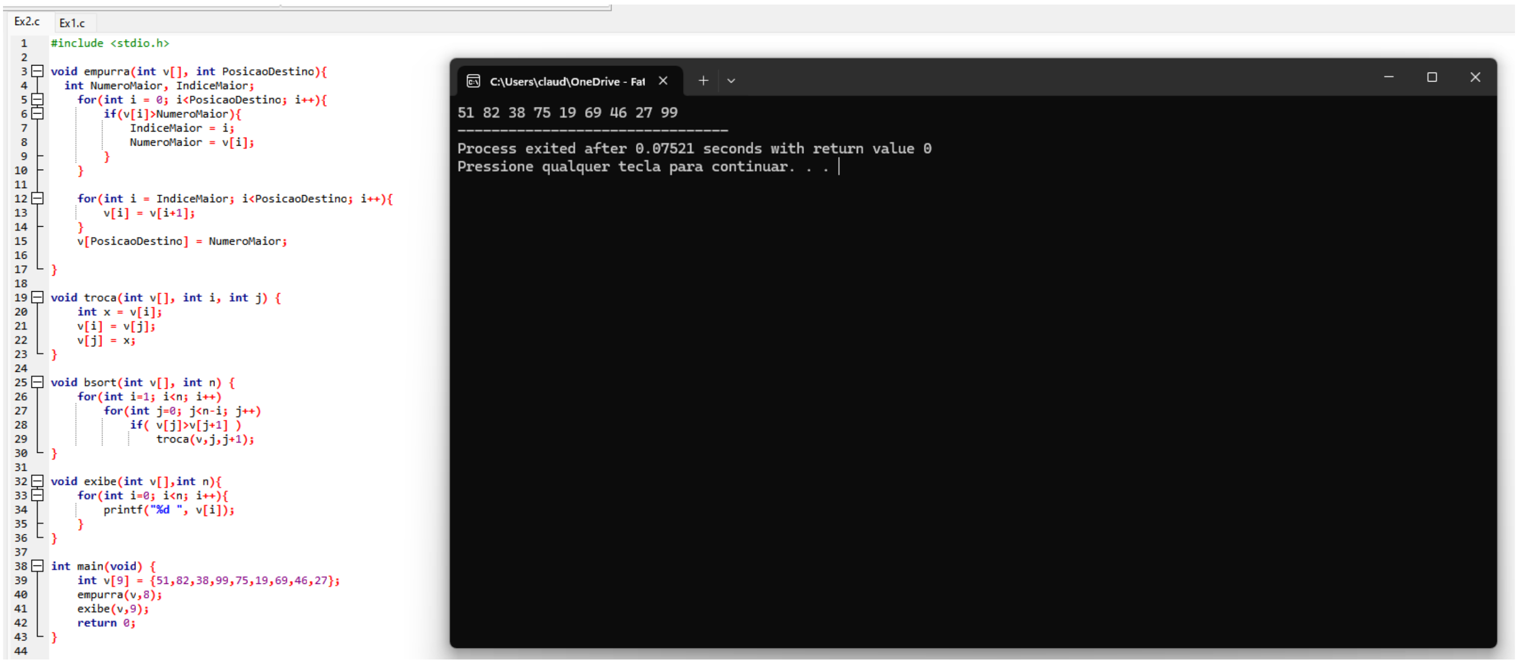
Exercício 1



```c
#include <stdio.h>
#include <stdlib.h>

void troca(int v[], int i, int j) {
    int x = v[i];
    v[i] = v[j];
    v[j] = x;
}

void bsort(int v[],int n) {
     for(int i=1; i<n; i++)
         for(int j=0; j<n-i; j++)
             if(v[j]>v[j+1] )
                 troca(v,j,j+1);
}

void exibe(int v[],int n){
     for(int i=0; i<n; i++){
         printf("%d ", v[i]);
     }
}

int main(void){
     int v[10] ={83,31,91,46,27,20,96,25,96,80};
     bsort(v,10);
     exibe(v,10);
     return 0;
}
```

```
20 25 27 31 46 80 83 91 96 96
--------------------------------
Process exited after 0.03084 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

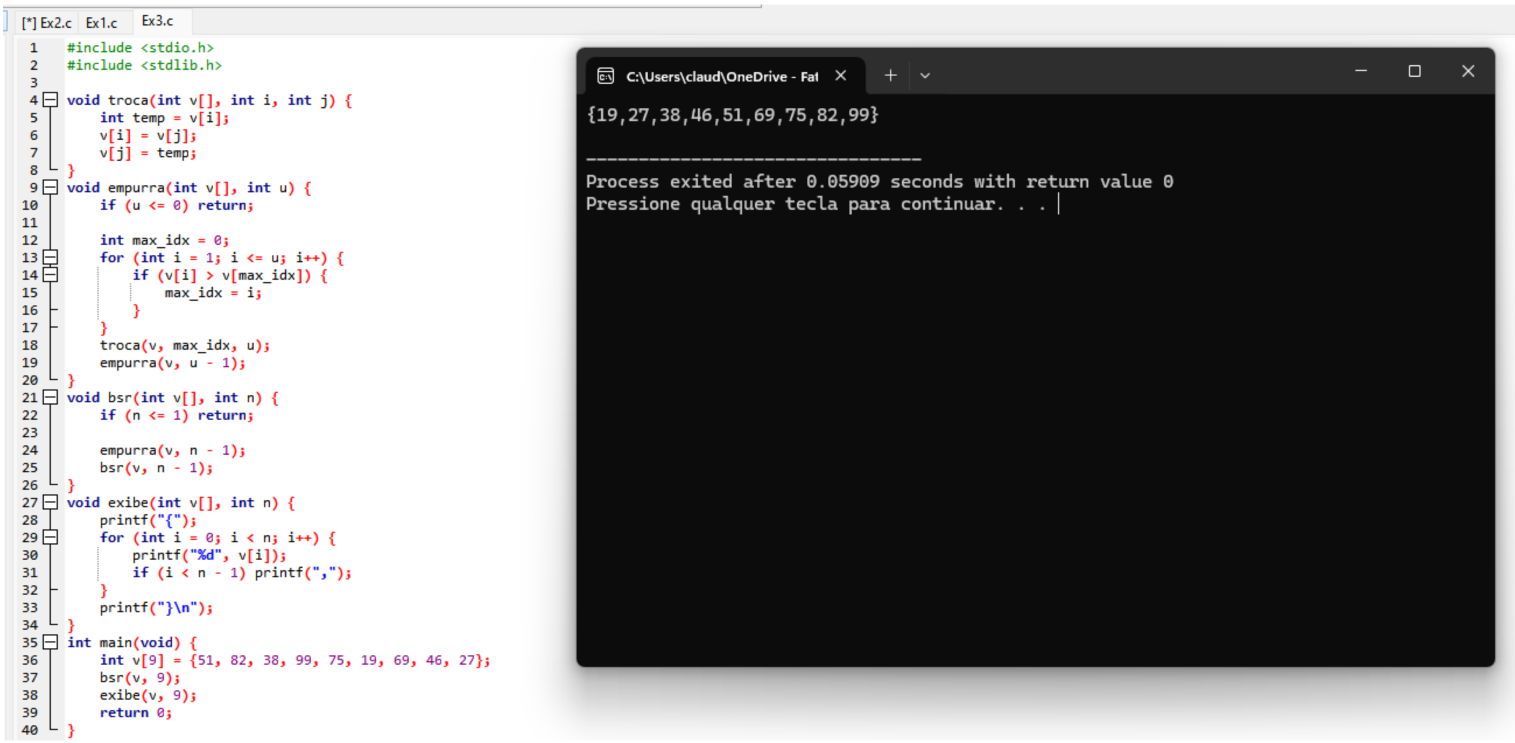
Exercício 2



```c
#include <stdio.h>

void empurra(int v[], int PosicaoDestino){
  int NumeroMaior, IndiceMaior;
    for(int i = 0; i<PosicaoDestino; i++){
        if(v[i]>NumeroMaior){
            IndiceMaior = i;
            NumeroMaior = v[i];
        }
    }

    for(int i = IndiceMaior; i<PosicaoDestino; i++){
        v[i] = v[i+1];
    }
    v[PosicaoDestino] = NumeroMaior;

}

void troca(int v[], int i, int j) {
    int x = v[i];
    v[i] = v[j];
    v[j] = x;
}

void bsort(int v[], int n) {
    for(int i=1; i<n; i++)
        for(int j=0; j<n-i; j++)
            if( v[j]>v[j+1] )
                troca(v,j,j+1);
}

void exibe(int v[],int n){
    for(int i=0; i<n; i++){
        printf("%d ", v[i]);
    }
}

int main(void) {
    int v[9] = {51,82,38,99,75,19,69,46,27};
    empurra(v,8);
    exibe(v,9);
    return 0;
}
```

```
51 82 38 75 19 69 46 27 99
--------------------------------
Process exited after 0.07521 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

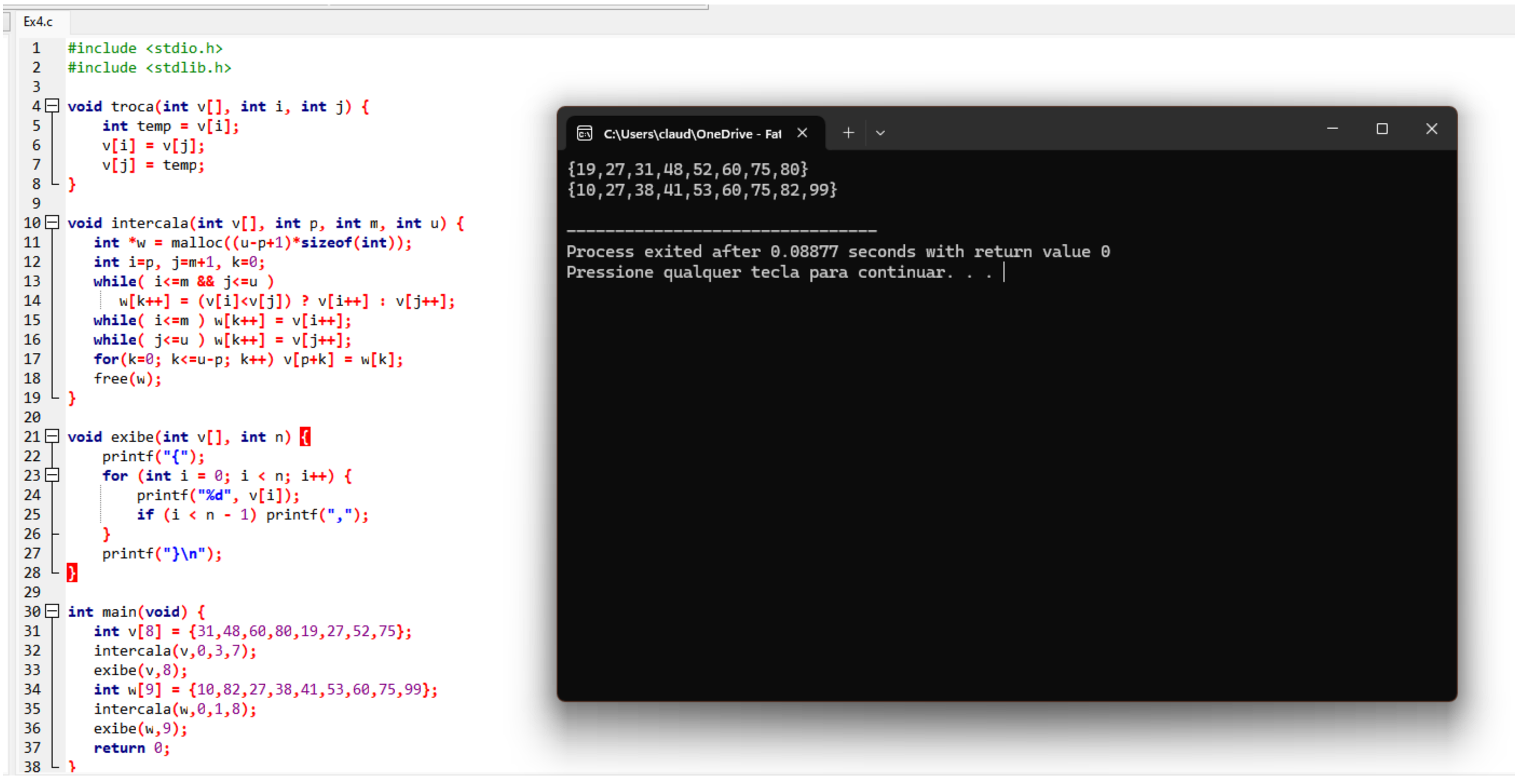
Exercício 3



```c
#include <stdio.h>
#include <stdlib.h>

void troca(int v[], int i, int j) {
    int temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}
void empurra(int v[], int u) {
    if (u <= 0) return;

    int max_idx = 0;
    for (int i = 1; i <= u; i++) {
        if (v[i] > v[max_idx]) {
            max_idx = i;
        }
    }
    troca(v, max_idx, u);
    empurra(v, u - 1);
}
void bsr(int v[], int n) {
    if (n <= 1) return;

    empurra(v, n - 1);
    bsr(v, n - 1);
}
void exibe(int v[], int n) {
    printf("{");
    for (int i = 0; i < n; i++) {
        printf("%d", v[i]);
        if (i < n - 1) printf(",");
    }
    printf("}\n");
}
int main(void) {
    int v[9] = {51, 82, 38, 99, 75, 19, 69, 46, 27};
    bsr(v, 9);
    exibe(v, 9);
    return 0;
}
```

```
{19,27,38,46,51,69,75,82,99}

--------------------------------
Process exited after 0.05909 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

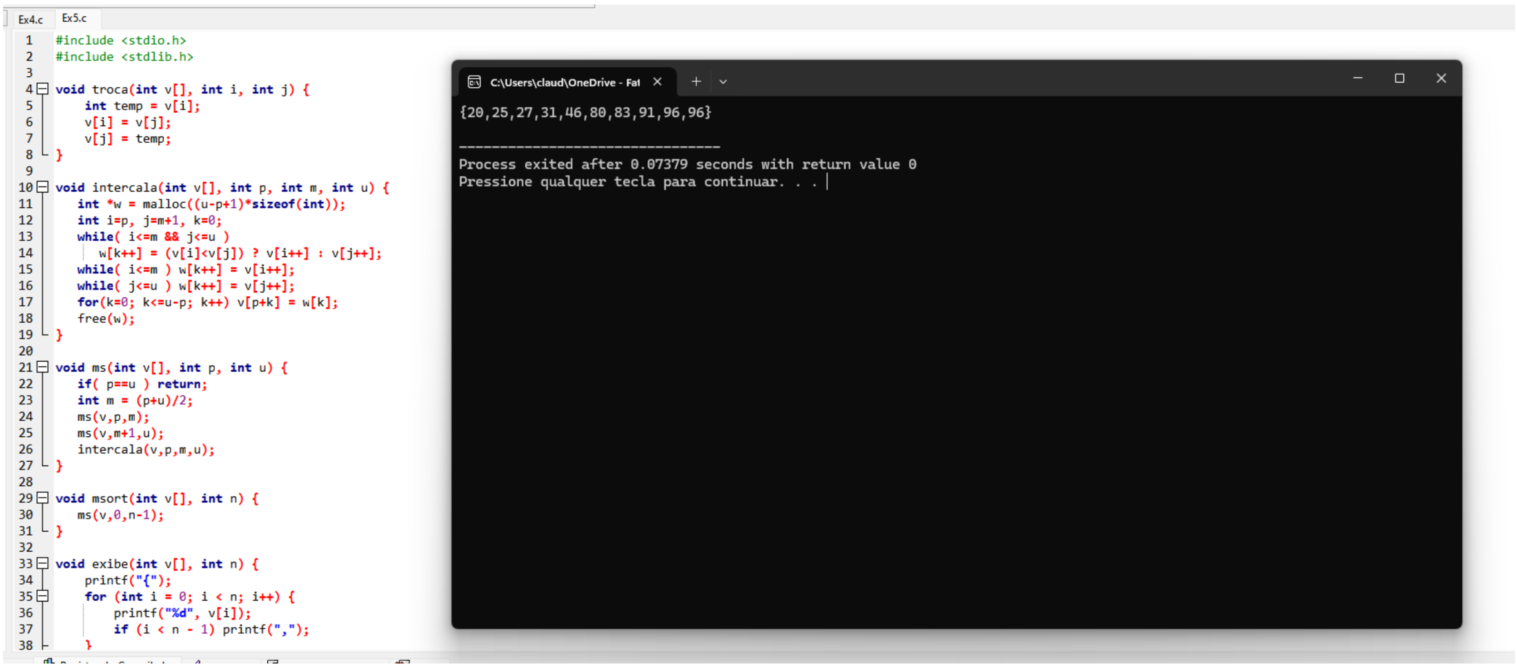
Exercício 4



```c
#include <stdio.h>
#include <stdlib.h>

void troca(int v[], int i, int j) {
    int temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}

void intercala(int v[], int p, int m, int u) {
    int *w = malloc((u-p+1)*sizeof(int));
    int i=p, j=m+1, k=0;
    while( i<=m && j<=u )
        w[k++] = (v[i]<v[j]) ? v[i++] : v[j++];
    while( i<=m ) w[k++] = v[i++];
    while( j<=u ) w[k++] = v[j++];
    for(k=0; k<=u-p; k++) v[p+k] = w[k];
    free(w);
}

void exibe(int v[], int n) {
    printf("{");
    for (int i = 0; i < n; i++) {
        printf("%d", v[i]);
        if (i < n - 1) printf(",");
    }
    printf("}\n");
}

int main(void) {
    int v[8] = {31,48,60,80,19,27,52,75};
    intercala(v,0,3,7);
    exibe(v,8);
    int w[9] = {10,82,27,38,41,53,60,75,99};
    intercala(w,0,1,8);
    exibe(w,9);
    return 0;
}
```

```
{19,27,31,48,52,60,75,80}
{10,27,38,41,53,60,75,82,99}

--------------------------------
Process exited after 0.08877 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

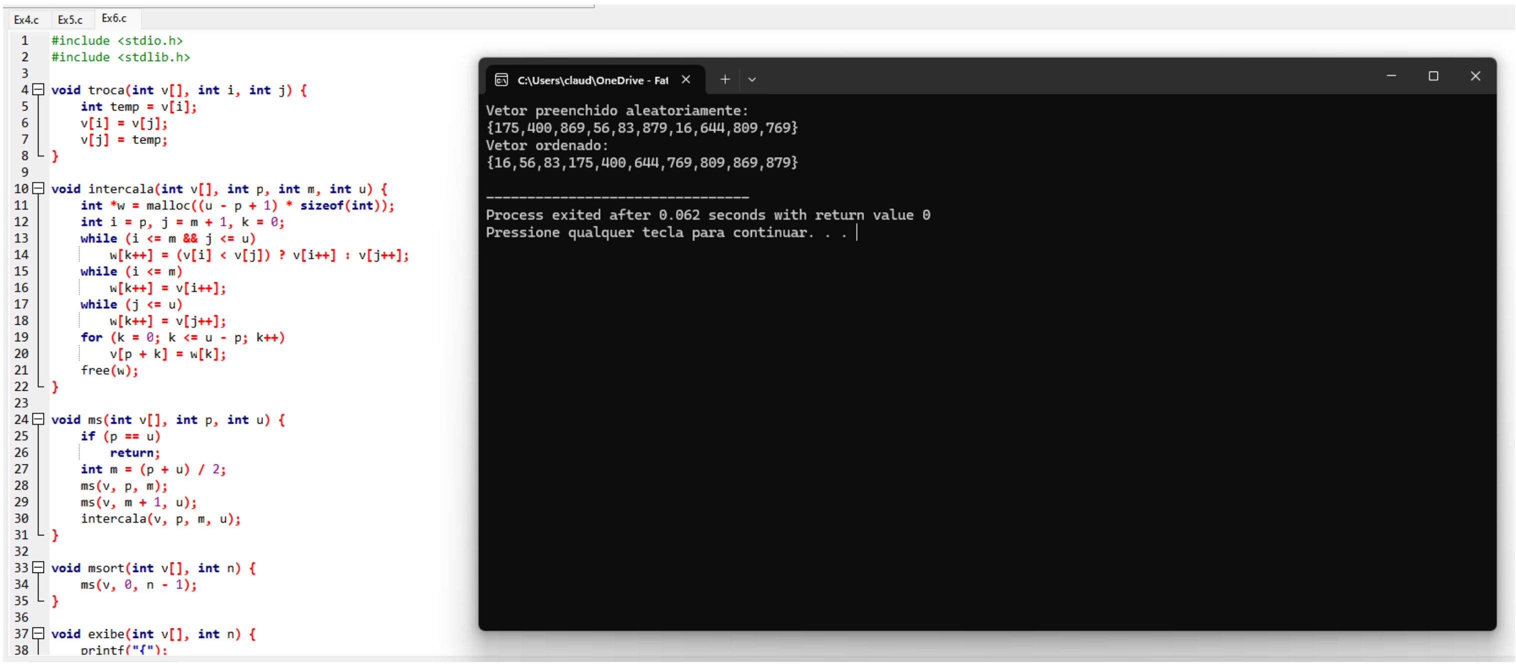
Exercício 5



```c
#include <stdio.h>
#include <stdlib.h>

void troca(int v[], int i, int j) {
    int temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}

void intercala(int v[], int p, int m, int u) {
   int *w = malloc((u-p+1)*sizeof(int));
   int i=p, j=m+1, k=0;
   while( i<=m && j<=u )
     w[k++] = (v[i]<v[j]) ? v[i++] : v[j++];
   while( i<=m ) w[k++] = v[i++];
   while( j<=u ) w[k++] = v[j++];
   for(k=0; k<=u-p; k++) v[p+k] = w[k];
   free(w);
}

void ms(int v[], int p, int u) {
   if( p==u ) return;
   int m = (p+u)/2;
   ms(v,p,m);
   ms(v,m+1,u);
   intercala(v,p,m,u);
}

void msort(int v[], int n) {
   ms(v,0,n-1);
}

void exibe(int v[], int n) {
    printf("{");
    for (int i = 0; i < n; i++) {
        printf("%d", v[i]);
        if (i < n - 1) printf(",");
    }
```

```
{20,25,27,31,46,80,83,91,96,96}

--------------------------------
Process exited after 0.07379 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

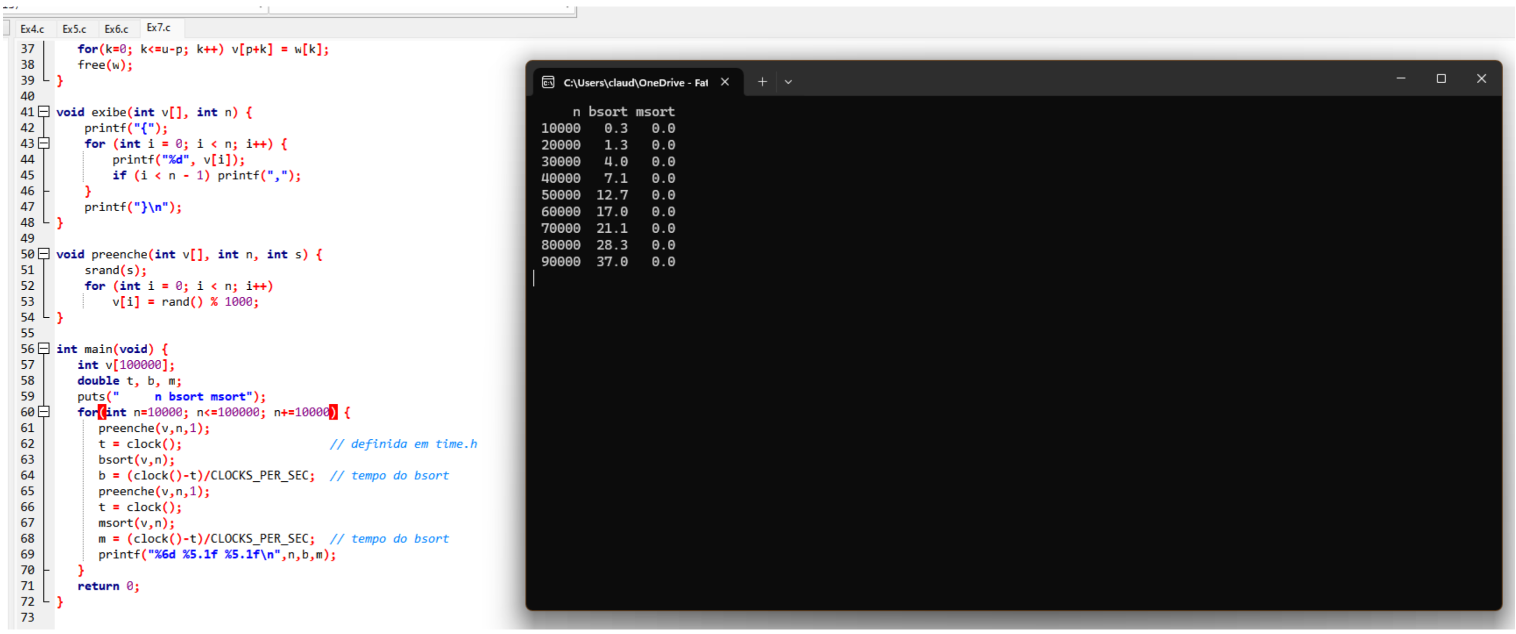
Exercício 6



```c
#include <stdio.h>
#include <stdlib.h>

void troca(int v[], int i, int j) {
    int temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}

void intercala(int v[], int p, int m, int u) {
    int *w = malloc((u - p + 1) * sizeof(int));
    int i = p, j = m + 1, k = 0;
    while (i <= m && j <= u)
        w[k++] = (v[i] < v[j]) ? v[i++] : v[j++];
    while (i <= m)
        w[k++] = v[i++];
    while (j <= u)
        w[k++] = v[j++];
    for (k = 0; k <= u - p; k++)
        v[p + k] = w[k];
    free(w);
}

void ms(int v[], int p, int u) {
    if (p == u)
        return;
    int m = (p + u) / 2;
    ms(v, p, m);
    ms(v, m + 1, u);
    intercala(v, p, m, u);
}

void msort(int v[], int n) {
    ms(v, 0, n - 1);
}

void exibe(int v[], int n) {
    printf("{");
```

```
Vetor preenchido aleatoriamente:
{175,400,869,56,83,879,16,644,809,769}
Vetor ordenado:
{16,56,83,175,400,644,769,809,869,879}

--------------------------------
Process exited after 0.062 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

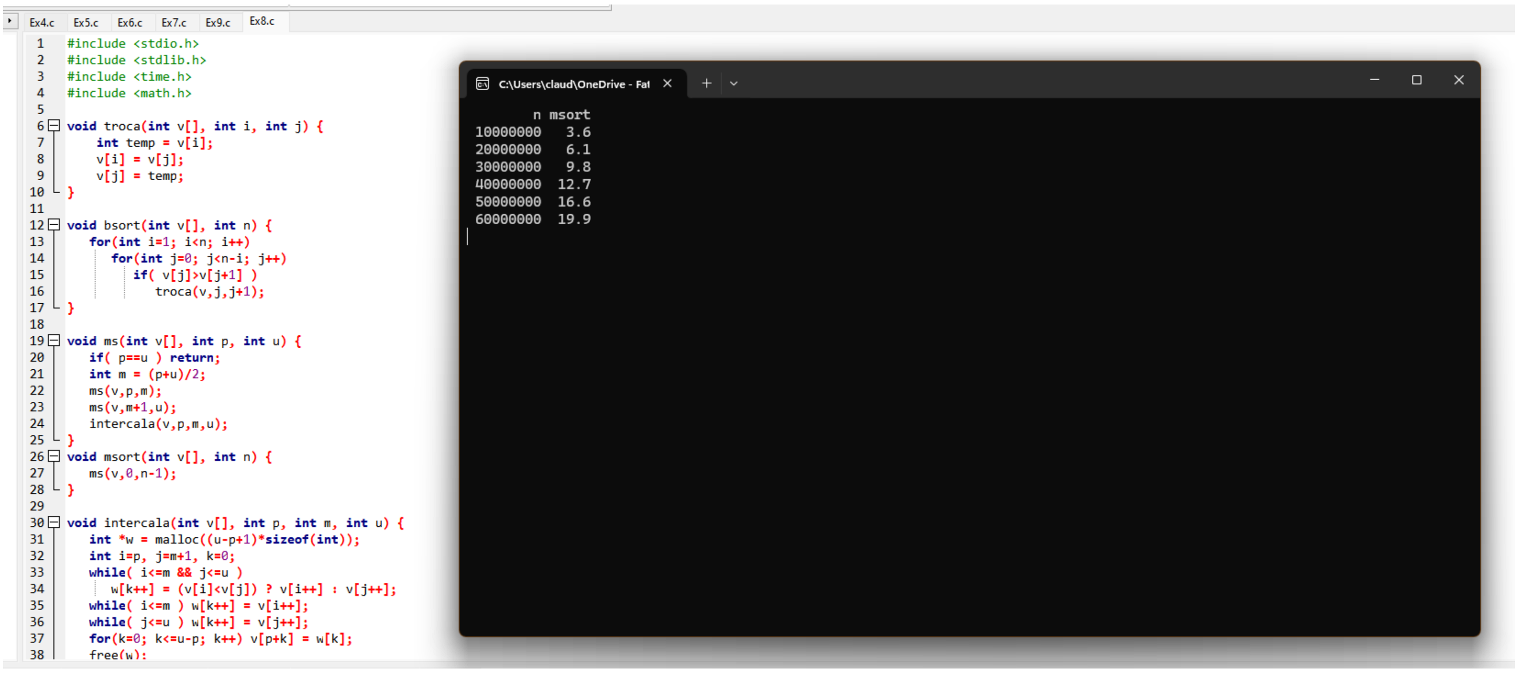
Exercício 7



```c
   for(k=0; k<=u-p; k++) v[p+k] = w[k];
   free(w);
}

void exibe(int v[], int n) {
    printf("{");
    for (int i = 0; i < n; i++) {
        printf("%d", v[i]);
        if (i < n - 1) printf(",");
    }
    printf("}\n");
}

void preenche(int v[], int n, int s) {
    srand(s);
    for (int i = 0; i < n; i++)
        v[i] = rand() % 1000;
}

int main(void) {
   int v[100000];
   double t, b, m;
   puts("     n bsort msort");
   for(int n=10000; n<=100000; n+=10000) {
      preenche(v,n,1);
      t = clock();                    // definida em time.h
      bsort(v,n);
      b = (clock()-t)/CLOCKS_PER_SEC;  // tempo do bsort
      preenche(v,n,1);
      t = clock();
      msort(v,n);
      m = (clock()-t)/CLOCKS_PER_SEC;  // tempo do bsort
      printf("%6d %5.1f %5.1f\n",n,b,m);
   }
   return 0;
}
```

```
     n bsort msort
 10000   0.3   0.0
 20000   1.3   0.0
 30000   4.0   0.0
 40000   7.1   0.0
 50000  12.7   0.0
 60000  17.0   0.0
 70000  21.1   0.0
 80000  28.3   0.0
 90000  37.0   0.0
```

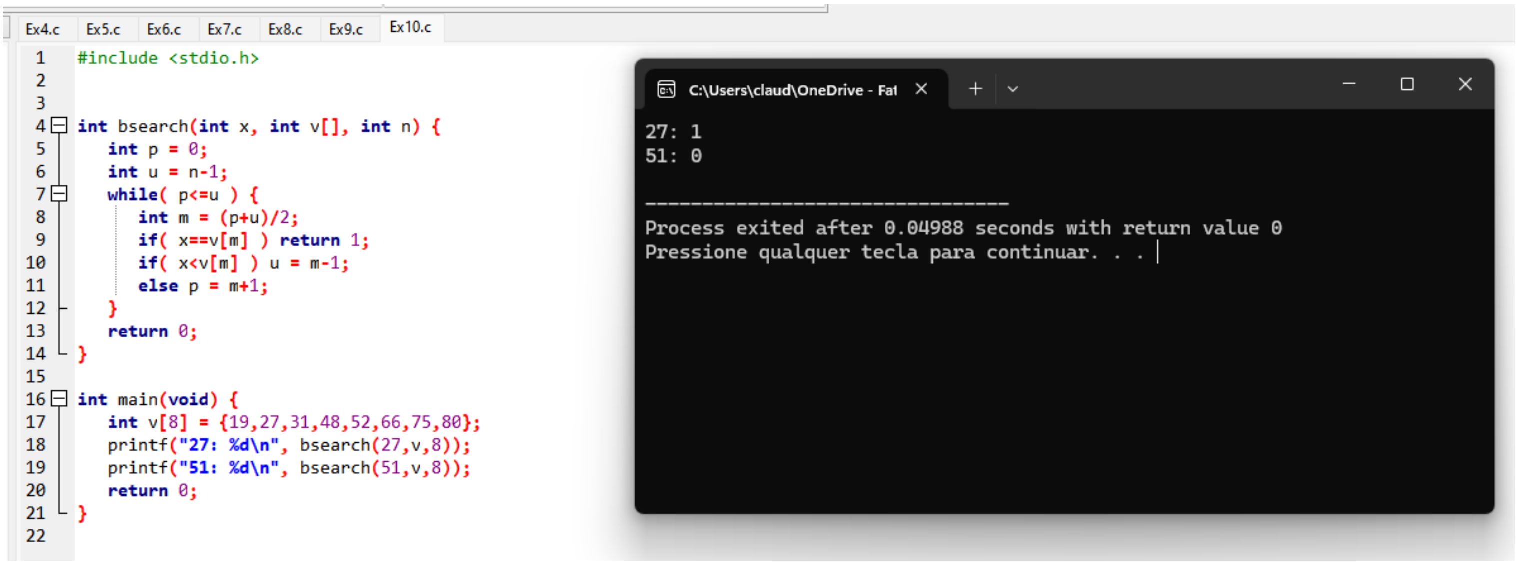
Exercício 8



```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

void troca(int v[], int i, int j) {
    int temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}

void bsort(int v[], int n) {
   for(int i=1; i<n; i++)
      for(int j=0; j<n-i; j++)
         if( v[j]>v[j+1] )
            troca(v,j,j+1);
}

void ms(int v[], int p, int u) {
   if( p==u ) return;
   int m = (p+u)/2;
   ms(v,p,m);
   ms(v,m+1,u);
   intercala(v,p,m,u);
}
void msort(int v[], int n) {
   ms(v,0,n-1);
}

void intercala(int v[], int p, int m, int u) {
   int *w = malloc((u-p+1)*sizeof(int));
   int i=p, j=m+1, k=0;
   while( i<=m && j<=u )
      w[k++] = (v[i]<v[j]) ? v[i++] : v[j++];
   while( i<=m ) w[k++] = v[i++];
   while( j<=u ) w[k++] = v[j++];
   for(k=0; k<=u-p; k++) v[p+k] = w[k];
   free(w);
```

```
       n msort
10000000   3.6
20000000   6.1
30000000   9.8
40000000  12.7
50000000  16.6
60000000  19.9
```

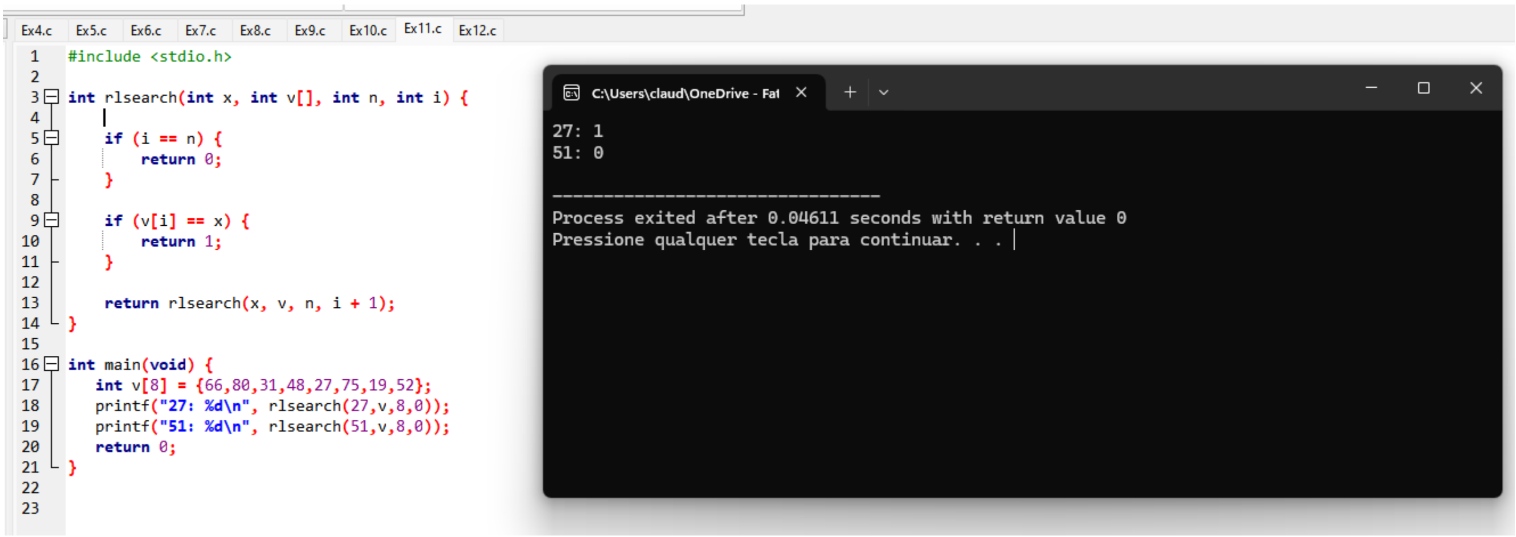
Exercício 10



```c
#include <stdio.h>


int bsearch(int x, int v[], int n) {
   int p = 0;
   int u = n-1;
   while( p<=u ) {
      int m = (p+u)/2;
      if( x==v[m] ) return 1;
      if( x<v[m] ) u = m-1;
      else p = m+1;
   }
   return 0;
}

int main(void) {
   int v[8] = {19,27,31,48,52,66,75,80};
   printf("27: %d\n", bsearch(27,v,8));
   printf("51: %d\n", bsearch(51,v,8));
   return 0;
}
```

```
27: 1
51: 0

--------------------------------
Process exited after 0.04988 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

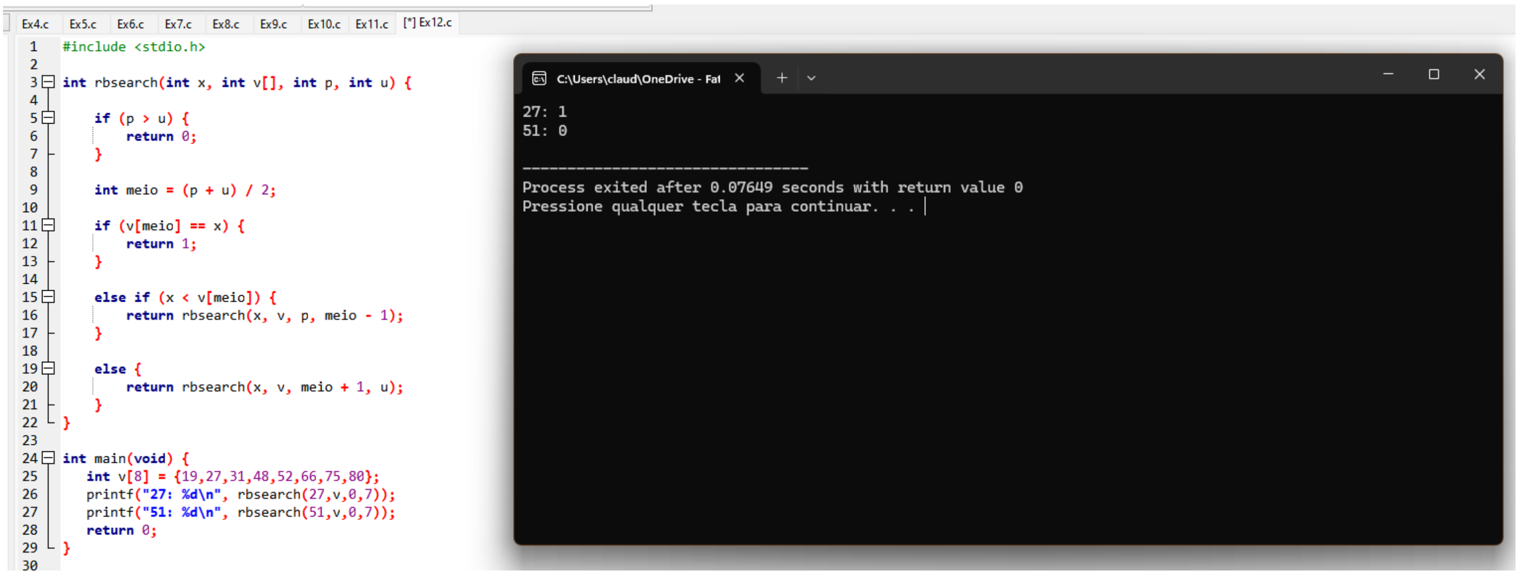
Exercício 11



```c
#include <stdio.h>

int rlsearch(int x, int v[], int n, int i) {
    
    if (i == n) {
        return 0;
    }

    if (v[i] == x) {
        return 1;
    }

    return rlsearch(x, v, n, i + 1);
}

int main(void) {
    int v[8] = {66,80,31,48,27,75,19,52};
    printf("27: %d\n", rlsearch(27,v,8,0));
    printf("51: %d\n", rlsearch(51,v,8,0));
    return 0;
}
```

```
27: 1
51: 0

--------------------------------
Process exited after 0.04611 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```

Exercício 12



```c
#include <stdio.h>

int rbsearch(int x, int v[], int p, int u) {

    if (p > u) {
        return 0;
    }

    int meio = (p + u) / 2;

    if (v[meio] == x) {
        return 1;
    }

    else if (x < v[meio]) {
        return rbsearch(x, v, p, meio - 1);
    }

    else {
        return rbsearch(x, v, meio + 1, u);
    }
}

int main(void) {
   int v[8] = {19,27,31,48,52,66,75,80};
   printf("27: %d\n", rbsearch(27,v,0,7));
   printf("51: %d\n", rbsearch(51,v,0,7));
   return 0;
}
```

```
27: 1
51: 0

--------------------------------
Process exited after 0.07649 seconds with return value 0
Pressione qualquer tecla para continuar. . .
```