Što će ispisati sljedeći program?

```
void pisi3(int *p, int i, int n) {
    if (i<n/2) {
        pisi3 (p,i+1,n);
        printf ("%d", *(p+i));
        printf ("%d", *(p+n-i-1));
    }
}
int main() {
    int p[4]={1,2,3,4};
    pisi3 (p,0,4);
    return 1;
}
```

4321
1234
3241
2314
1423

Funkcija za dodavanje elementa na stoga realiziran listom glasi:

```
int dodaj (int element, Stog *stog) {
   atom *novi;
   if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
       novi->element = element;
       novi->sljed = stog->vrh;
       stog->vrh = novi;
       return 1;
   }
   else return 0;
}
```

```
int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = element;
        return 1;
    }
    else return 0;
}
```

```
int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi->element = stog->vrh;
        novi->sljed = stog->vrh;
        return 1;
    }
    else return 0;
}
```

```
int dodaj (int element, Stog *stog) {
     atom *novi;
     novi->element = element;
     novi->sljed = stog->vrh;
     stog->vrh = novi;
     return 1;
}

 int dodaj (int element, Stog *stog) {
    atom *novi;
    if ((novi = (atom*) malloc(sizeof(atom))) != NULL) {
        novi = element;
        return 1;
    }
    else return 0;
}
```

Rekurzivna funkcija za izračunavanje n-tog Fibonnacijevog broja proširena je jednom naredbom za ispis te je njen kôd:

```
int F(int n) {
    if (n <= 1)
        return 1;
     else {
        printf("Test\n");
        return F(n-2) + F(n-1);
     }
}
```

Koliko puta će se na ekranu ispisati riječ "Test", ako se funkcija pozove s F(4)

3
4
1
7
9

Neka je zadana sljedeća rekurzivna funkcija:

```
int F(n) {
    if (n > 1)
        return F(n-2) + F(n-4) + F(n-6);
    else if (n==1)
        return 1;
    else
        return 0;
}
```

Koju vrijednost će funkcija vratiti u glavni program, ako je poziv funkcije bio F(4)

4
7
3
1
0

Funkcija za rekurzivno traženje broja unutar niza, zadana sljedećim kôdom:

```
int trazi (tip A[], tip x, int n, int i) {
  if(i >= n)
       return -1;
   if(A[i] == x)
       return i;
   return trazi (A, x, n, i+1);
}
```

ima složenost:

O(n)
O(log n)
O(n log n)
O(n2)
O(1)

Složenost Mergesorta je:

O(n log2n)
O(n3/2)
O(n2)
O(log2n)
O(n)

---

između 5 ponuđenih algoritama za uzlazno sortiranje niza
1, 3, 2, 5, 4, 7, 6,…,
999997, 999996, 999999, 999998, 1000000

najbolje je odabrati:

Poboljšani bubble sort
Shell sort sa Sedgwickovim slijedom: {1, 5, 19, 41, 109,…}
Mergesort
Shell sort s Hibbardovim slijedom: {1, 3, 7, …, 2k -1}
Selection sort

Prije početka sortiranja niza
9 1 5 8 3 6 7 2 4
QuickSortom, na temelju 3 elementa je procijenjen medijan
(prvi, posljednji, element na polovici) te je nakon toga stožer
stavljen na predzadnje mjesto. Kako izgleda niz u tom trenutku?
3 1 5 8 2 6 7 4 9
2 1 5 8 3 6 7 4 9
2 1 5 8 3 6 7 9 4
9 1 5 8 2 6 7 3 4

Nakon procjene medijana na temelju 3 elementa (prvi, posljednji i element na polovici)
te nakon zamjene mjesta stožera i predzadnjeg elementa, odmah na početku QuickSorta,
niz je sljedeći: 3 8 1 5 2 7 6 4 9. Kako izgleda niz nakon podjele na 2 podniza, koji se
rekurzivno sortiraju QuickSortom, i vraćanja stožera na svoje mjesto?

1 2 3 5 8 7 6 4 9
3 1 2 4 5 8 7 6 9
3 2 1 4 5 7 6 8 9
3 2 1 4 8 7 6 5 9
1 2 3 5 8 7 6 4 9

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije pop je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz pretpostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mjesta.

```
for (i=0; i<5; i++)
    push(i, &stog);

for (i=5; i>=0; i--)
    printf("%d ", pop(&stog));
```

0 1 2 3 4
4 3 2 1 0
5 4 3 2 1 0
4 3 2 1 0 -1
0 1 2 3 4 5

Funkcija push stavlja elemente na stog. Ako je operacija uspješno obavljena funkcija vraća 1, a u slučaju greške vraća 0. Prototip funkcije je:

```
int push (int element, Stog *stog);
```

Funkcija pop skida element sa stoga i vraća njegovu vrijednost ili -1 u slučaju greške. Prototip funkcije je:

```
int pop (Stog *stog);
```

Što će ispisati sljedeći programski odsječak, uz pretpostavku da je prije izvođenja stog prazan i da na njemu ima dovoljno mjesta.

```
printf("%d ", push(pop(&stog),&stog);
printf("%d ", pop(&stog));
```

0 -1
-1 1
1 -1
1 0
1 1

Između 5 ponuđenih algoritama za uzlazno sortiranje niza:
100000, 99999, 99998, 99997, 99996, 99995,…
15, 14, 13, 12, 11

najbolje je odabrati:

Shell sort sa Sedgwickovim slijedom: {1, 5, 19, 41, 109,…}
Merge sort
Poboljšani bubble sort
Shell sort s Hibbardovim slijedom: {1, 3, 7, …, 2k -1}
Selection sort