Вариант 3

№1 а

```
with recursive remployees(last_name,first_name,id, manager_id) as
  (select last_name,first_name, id, manager_id from bd6_employees where id = 1
   union all
   select bd6_employees.last_name,bd6_employees.first_name, bd6_employees.id,
bd6_employees.manager_id
     from bd6_employees join remployees on remployees.id =
bd6_employees.manager_id)
select * from remployees;
```

№1 б

```
create or replace procedure rec (id_man int) as $$
declare
        attr bd6_employees%ROWTYPE;
begin
        FOR attr IN
                SELECT * from bd6_employees WHERE
bd6_employees.manager_id=id_man ORDER BY id
        LOOP
        CALL rec(attr.id);

        RAISE INFO 'id: % ФИО:     % %     manager_id: %',
        attr.id,
        attr.last_name,
        attr.first_name,
        attr.manager_id;
        END LOOP;

end
```

```
$$ language plpgsql;

call rec(1);
```

№2

```
create or replace procedure cur () as $$
declare
    attr bd6_employees%ROWTYPE;
    my_cursor CURSOR FOR SELECT * FROM bd6_employees ORDER BY department_id,salary_in_euro;
    dep int;
    num int;
begin
    dep:=0;
    num:=0;
    OPEN my_cursor;
    LOOP
    FETCH my_cursor INTO attr;
    IF NOT FOUND THEN EXIT;
    END IF;
    IF dep!=attr.department_id THEN
    dep:=attr.department_id;
    num:=1;
    ELSE
    num:=num+1;
    END IF;
    attr.phone_number:=cast(attr.phone_number || ' Доб.'|| num as varchar);
    RAISE INFO '% ФИО:  % % %         ТЕЛЕФОН:         %',
    attr.department_id,
    attr.last_name,
    attr.first_name,
    attr.salary_in_euro,
    attr.phone_number;
    END LOOP;
    CLOSE my_cursor;

end
$$ language plpgsql;

call cur();
```

№3

```
CREATE OR REPLACE PROCEDURE upd() AS $$
DECLARE
  attr record;
BEGIN
  FOR attr IN
    (SELECT b6.id, b.manager_id as m2
      from bd6_employees b6 join (select b1.id, b2.manager_id
                                                                                  from bd6_employees b1 join bd6_employees b2 on b1.manager_id = b2.id) b
       on b6.id = b.id where b6.manager_id != 1 order by b6.manager_id, b6.salary_in_euro OFFSET 3)
  LOOP
    update bd6_employees  set manager_id = attr.m2 where id = attr.id;
  END LOOP;
END
```

```
$$ LANGUAGE plpgsql;

call upd();

SELECT * from bd6_employees;
```

№4

```
DROP TABLE IF EXISTS five;
CREATE TABLE five(f1 varchar ,f2 varchar,f3 varchar,f4 varchar, f5 varchar);
SELECT * FROM five;

create or replace procedure fiveW () as $$
declare
i int;
napravlenie int;
x1 int;
x2 int;
begin
        i:=1;
        x1:=0;
        x2:=0;
        napravlenie:=0;
        WHILE i<5000 LOOP
--      x1:=x1 mod 5 +1;
--      x2:=x1 mod 5 +1;
        CASE napravlenie
        WHEN 0 THEN
        INSERT INTO five VALUES (i,i+1,i+2,i+3,i+4);
        napravlenie:=1;
        i:=i+5;
```

```
WHEN 1 THEN
INSERT INTO five VALUES (i+4,i+3,i+2,i+1,i);
napravlenie:=0;
i:=i+5;
END CASE;

IF x1=0 OR x1=4 THEN
    if (napravlenie=1) THEN
        UPDATE five SET f1 = 'x' WHERE f1=cast((i-5) as varchar);
        UPDATE five SET f5 = 'x' WHERE f5=cast((i-1) as varchar);
    ELSE
        UPDATE five SET f1 = 'x' WHERE f1=cast((i-1) as varchar);
        UPDATE five SET f5 = 'x' WHERE f5=cast((i-5) as varchar);
    END IF;
END IF;
IF x1=1 or x1=3 THEN
    if (napravlenie=1) THEN
        UPDATE five SET f2 = 'x' WHERE f2=cast((i-4) as varchar);
        UPDATE five SET f4 = 'x' WHERE f4=cast((i-2) as varchar);
    ELSE
        UPDATE five SET f2 = 'x' WHERE f2=cast((i-2) as varchar);
        UPDATE five SET f4 = 'x' WHERE f4=cast((i-4) as varchar);
    END IF;
END IF;
IF x1=2 THEN
    UPDATE five SET f3 = 'x' WHERE f3=cast((i-3) as varchar);
END IF;

x1:=(x1+1) % 5;
```

```
        END LOOP;
end
$$ language plplgsql;

CALL fiveW();
SELECT * FROM five;
```