1)

```
create or replace function q1(i integer) returns void as $$
declare
z INTEGER;
a text[] := string_to_array('A B C D E F G H I J K L M N O P Q R S T U V W X Y Z', ' ');
begin
for z in reverse i..1 loop
raise notice '%',a[0:z];
end loop;
end; $$ Language plpgsql ;
```

2)

```
create or replace function two(s text) returns void as $$
declare
z INTEGER := 0;
a text[] := string_to_array(s, ' ');
i INTEGER := array_length(a,1);
-- ['A','B','C','D','E'];
begin
for z in 1..i loop
raise notice '%',a[0:z];
end loop;
end; $$ Language plpgsql ;
```

3)

```
CREATE OR REPLACE FUNCTION three(customer_type TEXT DEFAULT 'individual customer',
spend INTEGER DEFAULT 0,booked_holiday BOOLEAN DEFAULT FALSE ) RETURNS FLOAT
AS $$
BEGIN
IF customer_type = cast ('account customer' as text) THEN
IF spend > 25000 THEN
return 25;
ELSE return 10;
END IF;
ELSIF customer_type = cast ('individual customer' as text) THEN
IF booked_holiday THEN
RETURN 5;
ELSE RETURN 0;
END IF;
ELSE RETURN 0;
```

```
END IF;
END
$$ Language plpgsql;
```

4)

```
CREATE OR REPLACE FUNCTION steel_test(carbon_content float,rockwell_hardness integer,tensile
integer) returns integer as $$
BEGIN
IF carbon_content < 0.7 THEN
IF rockwell_hardness > 50 THEN
IF tensile > 30000 THEN RETURN 10;
ELSE RETURN 9;
END IF;
ELSE RETURN 8;
END IF;
ELSE RETURN 7;
END IF;
END
$$ language plpgsql;
```

5)

```
CREATE OR REPLACE FUNCTION insurance (income INTEGER, gender TEXT, age INTEGER,
married BOOLEAN ) RETURNS BOOLEAN AS $$
BEGIN
IF age > 30 AND (gender = 'M' OR married = TRUE) THEN RETURN TRUE;
ELSIF married = TRUE THEN
IF gender = 'M' AND income > 20000 THEN RETURN TRUE;
ELSIF gender = 'F' AND income < 20000 THEN RETURN TRUE;
ELSE RETURN FALSE;
END IF;
ELSE RETURN FALSE;
END IF;
END
$$ Language plpgsql;
```

6)

```
CREATE OR REPLACE FUNCTION extract_date(d DATE) RETURNS void AS $$
DECLARE
week_days text[] := Array['Sunday','Monday','Tuesday','Wednesday','Thrusday','Friday','Saturday'];
day double precision := EXTRACT(ISODOW FROM d);
month double precision := EXTRACT(ISOMONTH FROM d);
```

```
year double precision := EXTRACT(ISOYEAR FROM d);
BEGIN
raise notice 'Day of week: % \nMonth % \n Year %',day,month, year;
END
$$ Language plpgsql;
```

7)

```
CREATE OR REPLACE FUNCTION palindrome (num TEXT) RETURNS BOOLEAN AS $$
DECLARE
n_rev TEXT ;
BEGIN
num := cast (num as integer);
n_rev := reverse(num);
IF n_rev = num THEN RETURN TRUE;
ELSE RETURN FALSE;
END IF;
END
$$ Language plpgsql;
```

8)

```
CREATE OR REPLACE FUNCTION fibonacci (n INTEGER) RETURNS VOID AS $$
DECLARE
   num1 INTEGER := 0;
   num2 INTEGER := 1;
   num3 INTEGER := 0;
i INTEGER ;
BEGIN
raise notice '%' , num1;
raise notice '%', num2;
for i in 2..n loop
num3 := num1 + num2;
raise notice '%',num3;
num1 := num2;
num2 := num3;
end loop;
END
$$ language plpgsql;
```

9)

```
CREATE OR REPLACE FUNCTION consonant(s VARCHAR[]) RETURN VOID AS $$
DECLARE
   i INTEGER :=0;
   Count INTEGER :=0;
```

```
BEGIN
        FOR counter IN 1..n LOOP
          IF s[i] !=’a’ OR s[i]!=’e’ OR s[i]!=’i’ ORs[i]!=’o’ OR s[i]!=’u’ THEN
            Count++;
          END IF;
       END LOOP;
        RAISE NOTICE ‘ % ‘, Count;
END;
$$ language psql;
```

10)

```
CREATE OR REPLACE FUNCTION salary(bs BIGINT) RETURN VOID AS $$
DECLARE
Total_allowance INTEGER ;
total_deductions INTEGER ;
net_salary INTEGER;
DA FLOAT;
HRA FLOAT;
PF FLOAT;
PT FLOAT;
MA INTEGER := 500;
TA INTEGER := 2000;
BEGIN
         DA=0.53*bs;
        HRA=0.2*(bs+DA);
        PF=0.1*(bs+DA);
        IF bs>70000 THEN PT=200;
        ELSE IF bs<=70000 AND bs>50000 THEN PT=100;
        ELSE IF bs<=50000 AND bs>20000 THEN PT=50;
        ELSE IF bs<=20000 AND bs>10000 THEN PT=20;
        END IF;
        Total_allowance=DA+HRA+MA+TA;
        total_deductions = PF+PT;
        net_salary =bs+Total_allowance+total_deductions;
        Raise notice ‘Net salary : %’,net_salary;
END
$$ language psql;
```