1. Proszę pokazać przeplot operacji dwóch współbieżnych wątków, w wyniku którego współdzielona zmienna *n* osiągnie wartości:
   (a) 2 (b) 1 (c) 0 (d) –1 (e) –2

wątek A	wątek B
shared *n*: Integer := 0	
local *i*: Integer	**local** *j*: Integer
i := *n*	*j* := *n*
i := *i* + 1;	*j* := *j* - 1;
n := *i*;	*n* := *j*;
i := *i* + 1;	*j* := *j* - 1;
n := *i*;	*n* := *j*;

2. Proszę pokazać taki przeplot operacji w poniższym fragmencie programu (przykład dla procesu p_i), który
   (a) doprowadzi do naruszenia warunku wzajemnego wykluczania,
   (b) nie doprowadzi do naruszenia warunku wzajemnego wykluczania.

```
shared numer: Integer;

repeat
   numer := i;
until numer = i;
sekcja krytyczna;
```

3. Proszę ocenić poniższe algorytmy pod kątem poprawności rozwiązania problemu wzajemnego wykluczania dwóch procesów p_i oraz p_j (przykład p_i przy założeniu $i = 0$, $j = 1$), przyjmując wartości początkowe *num* = *i*, *z*[0] = *false*, *z*[1] = *false*.

a)
```
shared num: Integer;
while num ≠ i do
   nic;
sekcja krytyczna;
num := j;
reszta;
```

b)
```
shared z: array[0..1]
          of Boolean;
z[i] := true;
while z[j] do nic;
sekcja krytyczna
z[i] := false;
reszta
```

c)
```
shared z: array[0..1]
          of Boolean;
while z[j] do nic;
z[i] := true;
sekcja krytyczna
z[i] := false;
reszta
```

4. Proszę wykazać, że przedstawiony poniżej algorytm wzajemnego wykluczania dwóch procesów p_i i p_j (przykład dla procesu p_i) jest niepoprawny.

```
shared znacznik: array [0..1] of Boolean;
shared numer: Integer;

numer := j;
znacznik[i] := true;
while znacznik[j] and numer ≠ i do
   nic;
sekcja krytyczna_i;
znacznik[i] := false;
reszta_i;
```

5. Proszę przedstawić taki przeplot operacji w poniższej modyfikacji algorytmu piekarni, powstałej przez usunięcie tablicy *wybieranie* i wszystkich operacji na niej, który wykaże naruszenie poprawności rozwiązania problemu wzajemnego wykluczania.

```
shared wybieranie: array [0..n-1] of Boolean;
shared numer: array [0..n-1] of Integer;

local  k: Integer;

wybieranie[i] := true;
numer[i] := max(numer[0],numer[1],...) + 1;
wybieranie[i] := false;
for k := 0 to n-1 do begin
   if k ≠ i then begin
      while wybieranie[k] do nic;
      while numer[k] ≠ 0 and
            (numer[k],k)<(numer[i],i) do
         nic;
   end;
end;
sekcja krytyczna_i;
numer[i] := 0;
reszta_i;
```

6. Proszę dokonać analizy poprawności rozwiązania problemu wzajemnego wykluczania w poniższej modyfikacji algorytmu piekarni, powstałej poprzez usunięcie identyfikatorów procesów, jako mniej znaczącej składowej przy porównaniu numerów, nadanych w wejściu.

```
shared wybieranie: array [0..n-1] of Boolean;
shared numer: array [0..n-1] of Integer;

local  k: Integer;

wybieranie[i] := true;
numer[i] := max(numer[0],numer[1],...) + 1;
wybieranie[i] := false;
for k := 0 to n-1 do begin
   if k ≠ i then begin
      while wybieranie[k] do nic;
      while numer[k] ≠ 0 and
            (numer[k],k)<(numer[i],i) do
         nic;
   end;
end;
sekcja krytyczna_i;
numer[i] := 0;
reszta_i;
```

7. Proszę wykazać, że w algorytmie piekarni numery przydzielane procesom w przejściu przez drzwi mogą rosnąć w nieskończoność. Proszę wskazać taki przypadek, w którym numeracja ponownie zaczyna się od 1.

8. Proszę wykazać, że niepodzielność operacji *test&set* jest warunkiem koniecznym poprawności przedstawionego poniżej algorytmu wzajemnego wykluczania.

```
shared zamek: Boolean;

while test&set(zamek) do
   nic;
sekcja krytyczna;
zamek := false;
reszta;
```

9. Proszę wykazać, że niepodzielność operacji *exchange* jest warunkiem koniecznym poprawności przedstawionego poniżej algorytmu wzajemnego wykluczania.

```
shared zamek: Boolean;
local klucz: Boolean;

klucz =: true;
repeat exchange(zamek, klucz);
until klucz = false;
sekcja krytyczna;
zamek := false;
reszta;
```

10. Proszę opracować rozwiązanie problemu wzajemnego wykluczania z użyciem operacji *test&set* (ewentualnie dodatkowych zmiennych współdzielonych), gwarantujące ograniczone czekanie.

11. Proszę opracować rozwiązanie problemu wzajemnego wykluczania z użyciem operacji *exchange* (ewentualnie dodatkowych zmiennych współdzielonych), gwarantujące ograniczone czekanie.