Zadanie 1 (Semafory: Grzybobranie, 10p.) Przedsiębiorstwo **Knieja** prowadzi punkt skupu grzybów. Dysponuje on `T` > 1 pojemnikami, w których klienci wykładają dostarczone przez siebie grzyby.

Każdy klient po grzybobraniu (predefiniowana funkcja `Grzyby grzybobranie()`) udaje się do punktu skupu. Ponieważ pewne gatunki grzybów są bardziej poszukiwane niż inne, klienci którzy je dostarczają są uprzywilejowani wobec pozostałych (o tym, czy jest uprzywilejowany klient dowiaduje się wywołując funkcję `int jakiKient(Grzyby grzyby)`, która zwraca `0` dla klienta zwykłego i `1` dla uprzywilejowanego). Klient może wyłożyć dostarczone przez siebie grzyby tylko wtedy, gdy są dostępne pojemniki mogące pomieścić je wszystkie. Liczbę potrzebnych pojemników klient ustala wywołując funkcję `int ilePojemnikow(Grzyby grzyby)`. Ponadto klient nieuprzywilejowany ma prawo wyłożyć swoje grzyby tylko wtedy, gdy żaden klient uprzywilejowany nie czeka na pojemniki.

Uwaga: W związku z priorytetami klientów dopuszczalne jest zagłodzenie klientów nieuprzywilejowanych. Nie jest natomiast dopuszczalne zagłodzenie w obrębie grupy. W przypadku klientów nieuprzywilejowanych oznacza to, że jeśli od pewnego momentu nie będzie klientów uprzywilejowanych, to każdy nieuprzywilejowany zostanie obsłużony. Priorytety klientów obowiązują tylko do momentu rozpoczęcia wykładania grzybów do pojemników.

Punkt skupu zatrudnia **jednego** taksatora, który ocenia przyniesione grzyby i określa kwotę, którą należy wypłacić klientowi. Po wyłożeniu grzybów do pojemników klient czeka na taksatora. Taksator dokonuje wyceny wyłożonych przez klienta grzybów i przekazuje jej wynik klientowi, który na ten wynik czeka.

Następnie klient udaje się do kasy i po odebraniu pieniędzy informuje taksatora, że transakcja została zakończona. Po zakończeniu transakcji taksator przekazuje grzyby do magazynu i zwalnia zajmowane przez nie pojemniki.

Schematy procesów klienta i taksatora są pokazane niżej. Zapisz pełną treść procesów **Klient** i **Taksator** używając do synchronizacji semaforów binarnych i/lub ogólnych. Staraj się używać znaczących nazw zmiennych. Nie bój się komentować rozwiązania.

```
process Klient() {
  while (true) {
    Grzyby grzyby = grzybobranie();
    int u = jakiKlient(grzyby);
    int ile = ilePojemnikow(grzyby);
    // czeka na mozliwosc wylozenia grzybow
    wyloz(grzyby);
    // czeka na taksatora
    // czeka na wycene i informacje o kwocie do odebrania
    odbierzPieniadze(kwota);
    //zawiadamia taksatora o koncu transakcji
}}

process Taksator() {
  while (true) {
    // czeka az pojawia sie pojemniki klienta z grzybami do wyceny
    int kwota = wycen(grzyby);
    // powiadamia klienta o kwocie, czeka az klient zakonczy transakcje
    przekazDoMagazynu(grzyby);
    // zwalnia pojemniki klienta.
  }
}
```

Zadanie 2 (Monitory: Jaskinie, 10p.) W jaskini pracuje `G` > 0 grotołazów oraz **jeden** inspektor. Jaskinia składa się z `N` sal (`N` $\geqslant 3$) ponumerowanych od 0 do `N`-1 oraz `N` korytarzy łączących cyklicznie kolejne sale w kształt okręgu. Początkowo wszyscy znajdują się w sali o numerze 0.

Grotołaz cyklicznie:

- odpoczywa w sali, w której się znajduje, po czym decyduje, do której z dwóch sąsiednich sal pójdzie (predefiniowana funkcja `odpoczywamIWybieramSalę`)
- czeka na zgodę na przejście przez wybrany korytarz (funkcja monitorowa **chcęPrzejść(int sala, int następnaSala)**)
- przechodzi wybranym korytarzem (predefiniowana funkcja `przechodzę)`
- informuje, że przeszedł do docelowej sali (funkcja monitorowa **przeszedłem(int sala, int następnaSala)**)

Inspektor cyklicznie:

- decyduje, czy w aktualnej sali ma dokonać inspekcji, czy przechodzi do sąsiadującej sali (predefiniowana funkcja `coMamRobić`)
- jeśli zdecyduje się przejść do sąsiadującej sali, to podobnie jak grotołaz: czeka na zgodę na przejście danym korytarzem, przechodzi nim, a na koniec informuje, że przeszedł (funkcje monitorowe **chcęPrzejść(int sala, int następnaSala)** i **przeszedłem(int sala, int następnaSala)** oraz predefiniowana funkcja `przechodzę`)
- jeśli jednak zdecyduje się wykonać inspekcję, to: czeka na zgodę na jej wykonanie (funkcja monitorowa **chcęWykonaćInspekcję(int sala)**), robi inspekcję (predefiniowa funkcja `inspekcja`), po czym informuje, że zakończył prace (funkcja monitorowa **skończyłemInspekcję(int sala)**)

Ponieważ korytarze łączące sale są bardzo wąskie, w każdym momencie może się w nich poruszać grupa idąca tylko w jednym kierunku. Każda inspekcja wiąże się z pewnymi zagrożeniami, dlatego przez cały czas jej wykonywania, w sali, w której odbywa się inspekcja ani w żadnym z sąsiadujących korytarzy nie może przebywać żaden grotołaz.

Napisz kod monitora Jaskinia udostępniającego powyższe funkcje. Możesz założyć, że liczba `N` jest na tyle mała, że dopuszczalne jest korzystanie z dwuwymiarowych tablic `N` $\times$ `N`. Pamiętaj o zapewnieniu żywotności Grotołazów i Inspektora (przy założeniu, że żaden proces nie zawiesi się w funkcjach `odpoczywamIWybieramSalę` oraz `przechodzę`). Treść procesów Grotołaz i Inspektor jest następująca:

```
process Inspektor() {
 int sala = 0, nastepnaSala, akcja;
 while (true) {
  coMamRobic(&akcja, &nastepnaSala);
  if (akcja == IDE_DALEJ) {
   Jaskinia.chcePrzejsc(sala, nastepnaSala);
   przechodze(sala, nastepnaSala);
   Jaskinia.przeszedlem(sala, nastepnaSala);
   sala = nastepnaSala;
  } else { // akcja == INSPEKCJA
   Jaskinia.chceWykonacInspekcje(sala);
   inspekcja(sala);
   Jaskinia.skonczylemInspekcje(sala);
  }
 }
}
```

```
process Grotolaz() {
 int sala = 0, nastepnaSala;
 while (true) {
  odpoczywamIWybieramSale(sala, &nastepnaSala);
  Jaskinia.chcePrzejsc(sala, nastepnaSala);
  przechodze(sala, nastepnaSala);
  Jaskinia.przeszedlem(sala, nastepnaSala);
  sala = nastepnaSala;
 }
}
```