**Adam Zieliński 132349**

**Marcin Rochowiak 132313**

**Nanozombie**

Archeolodzy odkryli starożytne, podwodne miasto, opanowane całkowicie przez chmury nanitów, które po wykryciu obecności ludzi przybierają kształt zombie. Ponieważ miasto położone jest na międzynarodowych wodach, każdy może zostać turystą i zwiedzić miasteczko. Potrzebne jest do tego: strój kucyka (bo zombie, zmylone strojem, głupieją i zaczynają tańczyc i palić zioło), transport oraz osobisty sprzęt eksploracyjny. Transport to jedna z rozróżnialnych łodzi podwodnych, każda o innej pojemności. Każdy turysta musi zdobyć strój kucyka (nierozróżnialne) oraz dostęp do jednej z łodzi podwodnych. Każdy turysta zajmuje od 1 do n pojemności łodzi. Gdy łódź zostanie zajęta tak, że żaden inny turysta się nie zmieści (niekoniecznie optymalnie, rozważane jest wg kolejności zgłoszeń) wyrusza na wyprawę.

Zmienne każdego procesu: (dostęp z użyciem zegaru Lamporta)

Lodz kolejka\_lodzi[1..m1] (queue)

int lista\_turystów[1..m2] (lista)

int liczba\_strojow

int minimalna\_pojemnosc\_lodzi

struct Lodz{

int pojemnosc;

vector lista\_id\_turystow;

}

**Algorytm:**

1. USTAWIENIE ZMIENNYCH
   1. Losowanie liczby strojów kucyków i łodzi
   2. Przypisanie wszystkich Łodzi do **kolejki\_lodzi**.
2. PODZIAŁ NA PROCESY – TURYSTÓW
   1. Losowanie wagi (liczby) sprzętu eksploracyjnego.
3. Pobranie stroju
   1. Wysłanie żądania dostępu do stroju.
   2. Czekamy, aż otrzymamy odpowiedź (nie chce stroju, chce strój) od pozostałych procesów. Dokładnie potrzebujemy [liczba\_procesów] – [liczba\_strojów] odpowiedzi z informacją, że nie dany proces nie chce stroju lub ma większą wartość zegara Lamporta. Wtedy możemy pobrać strój, nawet kiedy nie mamy odpowiedzi od wszystkich.
4. ZAJĘCIE MIEJSCA W ŁODZI
   1. Wysłanie żądania dostępu do łodzi.
   2. Odbieranie odpowiedzi od pozostałych procesów (nie chce, chce transportu).
      1. Kiedy odbierzemy informacje od wszystkich, sortujemy listę turystów w kolejności zgodnej z zegarem Lamporta
      2. Zgodnie z listą turystów dokonujemy przydziału do łodzi (kiedy kolejny turysta się nie mieści przechodzimy do kolejnej łodzi) poprzez dopisanie procesu do **lista\_id\_turystow**.
      3. Pozostali turyści czekają na łódź w kolejce. Mogą zająć łódź kiedy wróci do portu.
5. Wyprawa
   1. Ostatni turysta na liście\_id\_turystów przejmuje kontrolę nad łodzią, losuje czas trwania wyprawy.
6. Zwolnienie łodzi
   1. Kiedy wyprawa dobiegnie końca, turysta sterujący łodzią wysyła do wszystkich infornację o powrocie łodzi oraz zwalnia łódź.
   2. Łódź można dodać do kolejki łodzi, kiedy pozostali sterujący łodziami odpowiedzą z informacją o swoim stanie (płyną, wrócili do portu). Dodajemy najpierw łodzie, które wróciły wcześniej. Dokładnie musimy otrzymać [liczba\_łodzi] – [liczba\_wolnych\_łodzi] odpowiedzi.
7. Nowy turysta
   1. Każdy proces, który zakończył podróż (dostał informację o powrocie lub sterował łodzią) zwalnia strój i wraca do punktu 2.