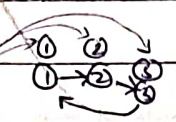
Hamiltonian cycle problem

Let G be a graph, find hamiltonian cycle. [start from vertex any A, travel all vertices once only, and return to vertex A again]



Approach

① f^n ① Hamiltonian(k)

in ∞ loop if x[k+1] is invalid we check next available vertex for position 'k'

1) call for nextValue(k), this will return what vertex to take for next step in array x[] at index k
2) if x[k]=0, then return, as we set nextValue(k) such that if there is no valid next vertex we set x[k]=0
3) if x k==n, ie we reach last vertex then we found our answer so write x[1:n], printing sequence of vertex for cycle
   else move for next sequence vertex Hamiltonian(k+1)

② f^n ② nextValue(k) doesn't matter if it is valid or not

in ∞ loop if vertex we choose for place x[k] turns out to be invalid then we check for next value of vertex through mod

1) choose next vertex using mod
2) if x[k]=0 then return, we exausted all vertices and none is valid.
3) if the vertex we chose from 2), have an edge with previous vertex in x[k-1] then do the following
   i] for loop from all elements in x till now, no vertex should be repeated in x
   ii] if above loop does not break, ie no other vertex in x has a edge with our vertex except the x[k-1]
   if [our vertex is not last] OR
   [our vertex is last and it has edge with 1st vertex in x]
   then return

G - Graph
x - array → index → sequence of cycle vertex, 1st vertex 2nd vertex, etc
value → which vertex?
K - current index
n - no. of vertices

```
x = [0]*n

Hamiltonian (K)
{
    repeat
    { nextValue (K)                    # what vertex to choose for index k in array x
      if (x[k]==0) then return         # no valid vertex found
      if (K== n) then write (x[1:n])   # we reach to all vertex
      else
          Hamiltonian (k+1)            # move for next vertex
    } until (false)
}

nextValue (k)
{
  repeat
    {                                  # iteratively choose next vertex
      x[k]= (x[k]+1) % (n+1)
      if (x[k]==0) then return         # exhausted all vertices
      if (G [x[k-1], x[k]]≠0) then     # if new vertex has edge with previous vertex
        { for j←1 to k-1 do            # loop through all vertices currently chosen in cycle
            if (x[j]==x[k])            # if any vertex is already chosen in array x then break
              then break

      if (j==k) then                   # if loop does not break
        if ((k<n) or                   # yet to travel all vertices
            ((k==n) and G[x[n],x[1]]≠0))
              then return              # reached all vertices then check if last vertice joins with 1st vertice to make cycle complete
    }
  } until (false)
}
```