**Logo

Description automatically generated San Francisco Bay University**

**CS350 - Data Structures**

**Homework Assignment #3**

**Due day: 3/1/2023**

**Instruction:**

**a. Push the source code to GitHub**

**b. Please follow the code style rule like programs on handout.**

**c. Overdue homework submission could not be accepted.**

**d. Take academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)**

1. In an open farmland where a hare (similar as rabbit) lives, it usually sleeps in any suitable place, continually shifting from one place to another in total *10* nests labeled from *1* to *10*.

But a wolf lives in the same area, hunting to check *10* nests in the manner as follows:

* start to check from label *1* nest
* then skip one nest (label *2*) to check label *3*
* increase skipped checking number to *2* (skip label *4* and label *5*) and look for it in label *6*
* keep increasing skipped number to *3* to check label *10*
* go back to count from label *1* by increasing skipped number to *4* and so on

Write a program to help this hare to make a decision which nests are safe to sleep, maybe doesn't exist at all after the wolf checked *n* times

Hint: create a **circular** linked list and traverse one by one circularly.

2. Assuming that there are two **circular** linked lists *l* & *m* with *char* type node value from

*a-zA-Z* in non-descending sequence, find a function/method to extract common node values from both and generate a new circular list without duplicated one.

Hint: take *question 4* in *HW#2* as reference

3. Given a **circular** linked list with *int* type value at each node, write a program to delete all prime number nodes, such as *m= Head->13->12->15->14->Head,* after calling ***delete\_prime\_CLL(m),*** you will get *Head->12->15->14->Head*

4. Find a program to split from first *N* nodes into new **circular** linked list with *int* type nodes while preserving the old nodes. For instance, *org = Head->2->3->4->5->6->7->8*

*->Head, two new* **circular** linked lists should be *Head->2->3->4->Head* and

*Head->5->6->7->8->Head* from the outputs of function call ***split\_CLL(org, 3)***

5. Generate a function in clockwise rotation for each node (either *char* type or *int* type) in **doubly** linked list by *N* places, e.g. given

*list = None<-Head<=>c<=>i<=>v<=>i>=>c->None,* from function call

***rotate\_DLL(list, 3)***,the output will be like this

*None<-Head<=>v<=>i<=>c<=>c>=>i->None*

6. (Bonus) Check whether there exists random pointer in **doubly** linked list or not and correct it by a program, as example as follows for function input and output

