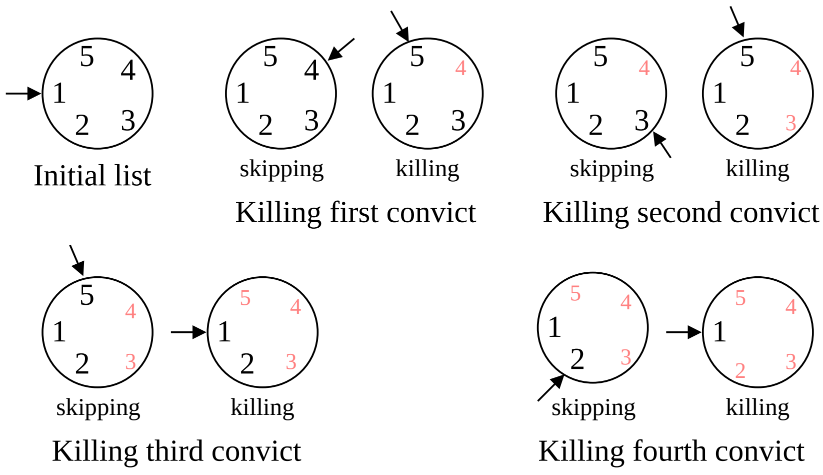
CS19002 PDS Lab, Spring 2007

Lab test 3 (Section 5)

Total points: 30 April 20, 2007 Time: 14:30–16:30

For students with odd PC numbers

[*Josephus problem*] Imagine a situation where n convicts numbered $1, 2, \ldots, n$ are sitting (in that order) at a round table in a prison. The prosecutor keeps on circling around the table in the order $1, 2, \ldots, n$ and then back to 1. He starts his rotation at convict 1, skips m living convicts, and kills the next living convict he encounters. After $n - 1$ iterations, only one convict survives and is set free. The following figure illustrates this situation with $n = 5$ and $m = 3$.



In this exercise, you write a program that, given $n \geqslant 0$ and $m \geqslant 0$, determines the survivor. You are asked to use a circular linked list which is like an ordinary linked list with the only exception that the pointer of the last node points to the first node (instead of being `NULL`). At every iteration of the loop, skip m nodes along the circular list and delete the $(m + 1)$-st node from the list. Imagine that after a convict is killed, his body is removed from the table. Note that when the head of the list is deleted, the head pointer needs to be adjusted accordingly.

Report the output of your program for the following values of (n, m). **(3 × 4)**

$n = 10, \quad m = 0.$

$n = 15, \quad m = 4.$

$n = 20, \quad m = 10.$

$n = 16, \quad m = 720720.$

Sample output

```
How many elements? 5
Enter skip amount: 3
Initial list: [ 1 2 3 4 5 ]
Deleting  4.  [ 1 2 3 5 ]
Deleting  3.  [ 1 2 5 ]
Deleting  5.  [ 1 2 ]
Deleting  2.  [ 1 ]
Survivor = 1
```

Follow the structure of the program given below.

```
#include <stdio.h>

typedef struct _node {
   int data;
   struct _node *next;
} node;

node *createList ( unsigned int n )
{
   /* Create a circular linked list with elements 1,2,3,...,n */                    (8)
}

void printList ( node *head )
{
   node *p;

   printf("[ ");
   if (head != NULL) {
      printf("%d ", head -> data);
      p = head -> next;
      while (p != head) {
         printf("%d ", p -> data);
         p = p -> next;
      }
   }
   printf("]\n");
}

node *deleteCyclic ( node *head, unsigned int m )
{
   /* Write this function */                                                          (10)
}

int main ()
{
   unsigned int n, m;
   node *head;

   printf("How many elements? "); scanf("%u",&n);
   head = createList(n);
   printf("Enter skip amount: "); scanf("%u",&m);
   printf("Initial list: "); printList(head);
   head = deleteCyclic(head,m); /* No need to pass n */
   if (head != NULL) printf("Survivor = %d\n", head -> data);
}
```

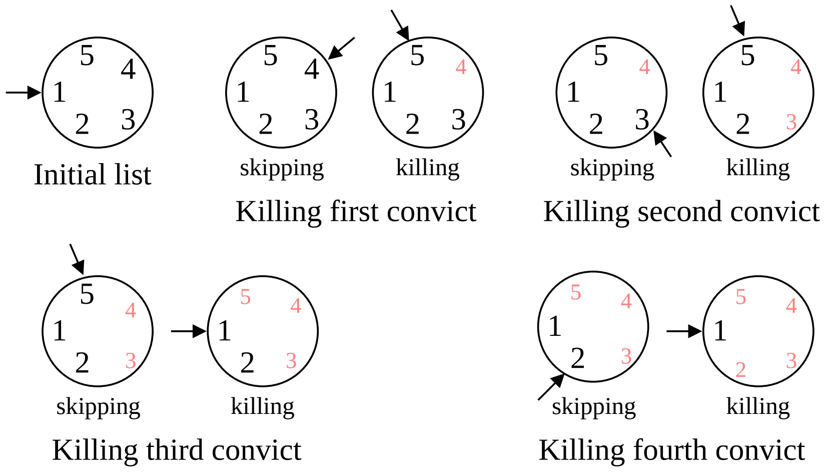
CS19002 PDS Lab, Spring 2007

Lab test 3 (Section 5)

Total points: 30 April 20, 2007 Time: 14:30–16:30

For students with even PC numbers

[*Josephus problem*] Imagine a situation where n convicts numbered $1, 2, \ldots, n$ are sitting (in that order) at a round table in a prison. The prosecutor keeps on circling around the table in the order $1, 2, \ldots, n$ and then back to 1. He starts his rotation at convict 1, skips m living convicts, and kills the next living convict he encounters. After $n - 1$ iterations, only one convict survives and is set free. The following figure illustrates this situation with $n = 5$ and $m = 3$.



In this exercise, you write a program that, given $n \geqslant 0$ and $m \geqslant 0$, determines the survivor. You are asked to use a circular linked list which is like an ordinary linked list with the only exception that the pointer of the last node points to the first node (instead of being `NULL`). Do not delete the nodes actually from the list. Imagine the situation that the dead bodies of the killed convicts continue to stay on the table. Initially, mark each node in the list as "LIVING". After a convict is killed, mark the corresponding node as "DEAD". When the prosecutor makes his circular trip around the table, he does not consider the convicts that are already dead. He skips m living convicts, and kills the $(m + 1)$-st living convict that he encounters.

Report the output of your program for the following values of (n, m). **(3 × 4)**

$n = 10, \quad m = 0.$

$n = 15, \quad m = 4.$

$n = 20, \quad m = 10.$

$n = 16, \quad m = 720720.$

Sample output

```
How many elements? 5
Enter skip amount: 3
Initial list: [ 1 2 3 4 5 ]
Deleting  4.  [ 1 2 3 5 ]
Deleting  3.  [ 1 2 5 ]
Deleting  5.  [ 1 2 ]
Deleting  2.  [ 1 ]
Survivor = 1
```

Follow the structure of the program given below.

```
#include <stdio.h>

#define LIVING 1
#define DEAD 0

typedef struct _node {
   int data;
   int mark;
   struct _node *next;
} node;

node *createList ( unsigned int n )
{
   /* Create a circular linked list with elements 1,2,3,...,n */                  (8)
   /* Mark each node as LIVING */
}

void printList ( node *head )
{
   node *p;

   printf("[ ");
   if (head != NULL) {
      if (head -> mark == LIVING) printf("%d ", head -> data);
      p = head -> next;
      while (p != head) {
         if (p -> mark == LIVING) printf("%d ", p -> data);
         p = p -> next;
      }
   }
   printf("]\n");
}

int deleteCyclic ( node *head, unsigned int n, unsigned int m )
{
   /* Write this function */                                                       (10)
}

int main ()
{
   unsigned int n, m;
   node *head;
   int survivor;

   printf("How many elements? "); scanf("%u",&n);
   head = createList(n);
   printf("Enter skip amount: "); scanf("%u",&m);
   printf("Initial list: "); printList(head);
   survivor = deleteCyclic(head,n,m);
   printf("Survivor = %d\n", survivor);
}
```