CSC10002 – Advanced Programming

Lab 03
Dynamic Arrays

Department of Software Engineering-FIT-VNU-HCMUS

1

Content

In this lab, we will review the following topics:

- How to use a pointer as a 1D array.
- How to use a pointer as a 2D array.

2 Assignments

A: YY: 01

H: YY: 05

2.1 Assignment – Replace String

Write function `char* replaceStr(char* src, char* sub, char* rep);` to replace all occurrences of sub-string `sub` in source string `src` with replacement string `rep`. Source string `src` must be unchanged.

Notice:

- The returned result string must be allocated dynamically.
- If source string src is NULL, the function returns NULL.

For example:

```
src = "brown fox and brown dog"
sub="brown"
rep="red"
Result string = "red fox and red dog"
```

2.2 Assignment – Pascal Triangle

Write function `int** createPascalTriangle(int N);` to allocate memory for a half matrix which stores a Pascal Triangle of N rows. The function returns a pointer to the allocated matrix whose elements are initialized based on the rule of Pascal Triangle.

```
Matrix(n, k) = Matrix(n - 1, k - 1) + Matrix(n - 1, k)
```

For example: Below is the Pascal Triangle with 5 rows

1				
1	1			
1	2	1		
1	3	3	1	
1	4	6	4	1

2.3 Assignment – Computer Labs

You run M computer labs. each lab contains Ni computer stations.

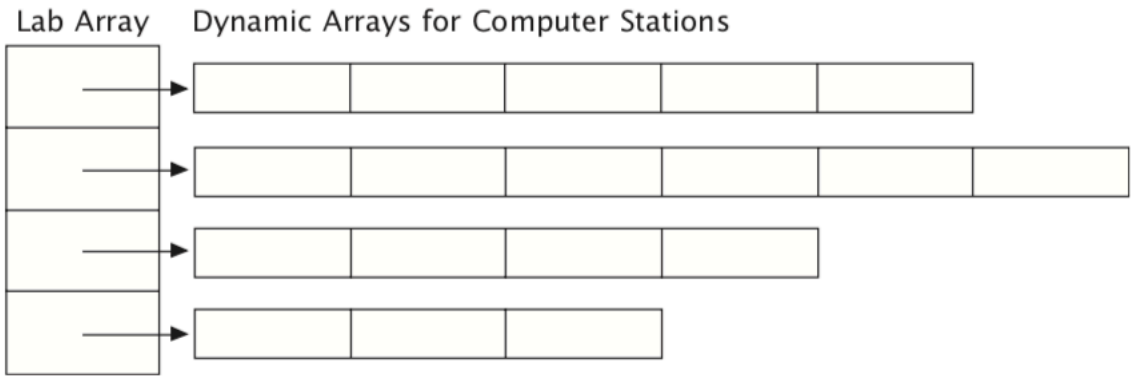
Each user has a unique five-digit ID number. Whenever a user logs on, the user's ID, lab number, and the computer station number are transmitted to your system. For example, if user 49193 logs onto station

2 in lab 3, then your system receives 49193 2 3 as input data. Similarly, when a user logs off a station, then your system receives the lab number and computer station number.

Write a computer program that could be used to track, by lab, which user is logged onto which computer. For example, if user 49193 is logged into station 2 in lab 3 and user 99577 is logged into station 1 of lab 4, then your system might display the following:

Lab	Station					
	1	2	3	4	5	6
1	0	0	0	0	0	
2	0	0	0	0	0	0
3	0	49193	0	0		
4	99577	0	0			

Create a menu that allows the administrator to simulate the transmission of information by manually typing in the login or logoff data. Whenever someone logs in or out, the display should be updated. Also write a search option so that the administrator can type in a user ID and the system will output what lab and station number that user is logged into, or “None” if the user ID is not logged into any computer station.



2.4 Assignment – Zoo

A local zoo wants to keep track of how many pounds of food each of its n monkeys eats each day during a typical week. Write a program that stores this information in a two-dimensional n × 7 array, where each row represents a different monkey and each column represents a different day of the week. The program should first have the user input the data for each monkey. Then it should create a report that includes the following information:

- Average amount of food eaten per day by the whole family of monkeys.
- The least amount of food eaten during the week by any one monkey.
- The greatest amount of food eaten during the week by any one monkey.

Input Validation: Do not accept negative numbers for pounds of food eaten.

2.5 Assignment - Ticket

Write a program that can be used by a small theater to sell tickets for performances. The theater's auditorium has 15 rows of seats, with 30 seats in each row. The program should display a screen that shows which seats are available and which are taken. For example, the following screen shows a chart depicting each seat in the theater. Seats that are taken are represented by an * symbol, and seats that are available are represented by a # symbol:

```
                          Seats
               123456789012345678901234567890
Row  1         ***###***###*########*****####
Row  2         ####*************####*******##
Row  3         **###**********########****###
Row  4         **######**************##******
Row  5         ********#####*********########
Row  6         ##############************####
Row  7         #######************###########
Row  8         ************##****############
Row  9         #########*****############****
Row  10        #####*************############
Row  11        #**********#################**
Row  12        #############********########*
Row  13        ###***********########**######
Row  14        ##############################
Row  15        ##############################
```

Here is a list of tasks this program must perform:

- When the program begins, it should ask the user to enter the seat prices for each row. The prices can be stored in a separate array. (Alternatively, the prices may be read from a file.)
- Once the prices are entered, the program should display a seating chart similar to the one shown above. The user may enter the row and seat numbers for tickets being sold. Every time a ticket or group of tickets is purchased, the program should display the total ticket prices and update the seating chart.
- The program should keep a total of all ticket sales. The user should be given an option of viewing this amount.
- The program should also give the user an option to see a list of how many seats have been sold, how many seats are available in each row, and how many seats are available in the entire auditorium.

Input Validation: When tickets are being sold, do not accept row or seat numbers that do not exist. When someone requests a particular seat, the program should make sure that seat is available before it is sold.