DO NOT OPEN THIS EXAM UNTIL TOLD TO DO SO.

The instructions below must be followed strictly. Failure to do so can result in serious grade loss.

⇒ You may not
• talk to anyone once the exam begins.
• leave the examination room and then return.

⇒ Keep your eyes on your own paper.

⇒ Read all questions very carefully before answering them.

Specific instructions:

2. Calculator usage: No Calculator Allowed
3. Write in pen/pencil: Any
4. Any other instruction(s): Make sure you have clearly written your roll number.

Q# | 1  | 2  | 3  | 4  | 5  | 6  | TOTAL |
---|----|----|----|----|----|----|-------|
Max Marks | 20 | 15 | 15 | 15 | 15 | 20 | 100    |

Marks Obtained

CS192 End Term Exam | 1 | February 16, 2004
1. Akar Bakar Medical Center offers three kinds of health coverage
   • Outpatient
   • Hospitalization without surgery
   • Hospitalization with surgery
A patient can have only one coverage at a given time.
Develop a set of structures that will store the following information about a patient:
   • Patient name
   • Patient address
   • Patient record number
   • For outpatients
     Date of visit, diagnosis, and medicine prescribed
   • For patients hospitalized
     Date of admission, name of ward
   • For patients admitted for surgery
     Date of surgery, type of surgery, name of surgeon

Do not use any absolute constants. Declare all constants as enum only.
Make appropriate assumptions, eg length of name, length of address, number of medicines that could be prescribed, date format, etc. Your program could be divided into the following.

   • Constants
   • Patient info
   • Outpatients
   • Hospitalized patients
   • Patients gone through surgery
   • Health coverage
   • Health record
Develop a data structure to store the information represented in a node of the list shown. Use appropriate conventions. Make appropriate assumptions for the size of the name.

Write a utility function called `getNode` that will:

- Allocate memory to one node
- Check if memory was allocated
- If no memory was allocated then
  - Generate appropriate error message and then exit.
- Otherwise
  - Properly initialize members of the structure and then
  - Return the address of the node just acquired

Make sure that the function gets and returns what is needed.
Write a function called `displayIno` that gets a constant pointer to the start of the list mentioned above and prints the data of each node in a tabular format i.e. one record per row.

Make appropriate assumptions for the size of the name.
3. Consider the following type definition:

```c
struct ShoeType
{
    char style;
    double price;
};
```

Write a function `initShoeType` that will accept a reference to a record of `ShoeType` and then will initialize the members to a known initial state of zero.

Write a function `setShoeType` that accepts a reference to the record `ShoeType`, and values for style and price. Then sets the members accordingly.

Write a function `discountShoeType` that accepts a copy of a record of `ShoeType`, discounts the price by 10% and returns a copy of the record.
4. Write a program that reads golf scores in an array and shows how much each differs from the average. The skeleton structure of the program is given below:

```c
#include <iostream.h>
const int MAX_NUMBER_SCORES = 10;

void fillArray(int a[], int size, int &numberused)
{
    // precondition: size is the declared size of the array a
    // postcondition: numberused is the number of values stored in a
    // a[0] through a[numberused-1] have been filled with
    // nonnegative integers read from the keyboard
}

double computAverage(const int a[], int numberused)
{
    // precondition: a[0] through a[numberused-1] have values
    // numberused > 0
    // returns the average of numbers a[0] through a[numberused-1]
}
void showDifference(const int a[], int numberused)
{
    // precondition: The first numberused indexed variables of a[] have values
    // postcondition: Gives screen output showing how much each of the
    // first numberused elements of a[] differ from their average
}

void main()
{
    int scores[MAX_NUMBER_SCORES]
    int numberused;

    cout << 'This program reads golf scores and shows
'     << "how much each differs from the average.\n"
    << "Enter golf scores:\n"
    fillArray(score, MAX_NUMBER_SCORES, numberused);
    showDifference(score, numberused);
}

Sample Dialogue

This program reads golf scores and shows how much each differs from the average.
Enter golf scores:
Enter up to 10 nonnegative whole numbers
Mark the end of the list with a negative number
69 74 68 –1
Average of the 3 scores = 70.3333
The scores are:
69 differs from average by –1.33333
74 differs from average by 3.66667
68 differs from average by –2.33333
5.  (a) Rewrite the following function using pointers only. The prototype of the new function is:

```c
int patternMatchingUsingPointers(char *str1, char *str2);
```

```c
int patternMatchingUsingArrays(char str1[], char str2[])
{
    printf("Function: patternMatchingUsingArrays\n");

    int j, k;

    int len1 = strlen(str1);
    int len2 = strlen(str2);

    for(j=0; j<len1; ++j)
    {
        for(k=0; k<len2; k++)
        {
            if(str2[k] != str1[k+j])
                break;
        }
        if(k==len2)
            return j;
    }
    return -1;
}
```
Write a function that reads the following information from a text file:

Number of rows of matrix A
Number of columns of matrix A
Elements of matrix A separated by spaces
Multiplier

The function then multiplies each element of matrix A with the multiplier, and stores the result in matrix B. The function also saves the following information of matrix B in another file:

Number of rows of matrix B
Number of columns of matrix B
Elements of matrix B in row major form

Declare all matrices dynamically. Use pointer notation only. Do not use array notation.

Example Input Data File

2
3
1 2 3 4 5 6
5

Example Output Data File

2
3
5 10 15
20 25 30
6. Write a program that can be used to gather statistical data about the number of movies college students see in a month. The program should perform the following steps:

- Ask the user how many students were surveyed. An array of integers with this many elements should then be dynamically allocated.
- Allow the user to enter the number of movies each student saw into the array. Calculate and display the average, median, and mode of the values entered.

Do not accept negative numbers for input.
Check if memory is allocated or not and take appropriate action.
Write separate functions to calculate average, median, and mode.

Average = sum of all number / number of elements
Mode = the value in the array that occurs most often. If none of the value occurs more than once, the mode is none, return −1.
Median = middle value out of a set of sorted values. If the set contains even number of values then the median is mean of the two middle values.

Please note that you need to write a sorting function as well.