The University of Nottingham Ningbo China

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

A LEVEL 1 MODULE, 2023-2024

EEEE1044 Introduction to Software Engineering and Programming

Time allowed: **TWO Hours**

Candidates may log in to computers, test CodeBlocks and sign their desk card, but must NOT write anything else until the start of the examination period is announced.

Answer ALL Questions

Only a calculator from approved list A (or one functionally equivalent) may be used in this examination.

Dictionaries are not allowed with one exception. Those whose first language is not English may use a dictionary to translate between that language and English provided that neither language is the subject of this examination.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

Do read the Exam Procedure available on desktop before starting the exam.

ADDITIONAL MATERIAL: Exam Procedure

INFORMATION FOR INVIGILATOR:
IT support before, during and after exam is requested.

Q1.

Marking scheme

Mark	Item
2	Comment
2	Indentation & blank lines
3	Read in 2 int and determine the smaller/larger int
2	Call external within for loop in main()
4	External function format, find factor
2	Display format (left-justified, spacing)

```
// Calculate factors between limits provided by user
#include <stdio.h>
void findFactors ( int number ); // function prototype
int main ( void )
{
 int limit1, limit2, midNumber, i;
 puts( "input two integers: " );
 scanf( "%d%d", &limit1, &limit2 ); // get 2 integers from the user
 if ( limit1 > limit2 )
 {
 midNumber = limit1;
 limit1 = limit2; // set limit1 to a smaller number
 limit2 = midNumber; // set limit2 to a larger number
 } // end if
 printf("%-10s%s\n", "Number", "Factors of this number" ); // display the table headings
 for ( i = limit1; i <= limit2; ++i ) // loop from limit1 to limit2
 {
 printf( "%-10d", i ); // display the number
 findFactors( i ); // call defined function
 puts(""); // start in a new line after finding all the factors for i
 } // end for
} // end main
// find and display the factors of an integer
void findFactors ( int number )
{
 int j; // counter in for loop
 for ( j = 1; j <= ( number / 2 ); ++j )
 {
 if ( ( number % j ) == 0 )
 {
 printf( "%-5d", j ); // display the factor
 } // end if
 } // end for
} // end defined function
```

Q2

Marking scheme

Mark	Item
2	Comment
2	Indentation & blank lines
2	Loop -1 to stop
3	Check valid input, invalid input not counted
2	Separate count of failed marks
2	Calculate result
2	User-friendly display

```
// Calculate and count mark of indefinite number of students
#include <stdio.h>
int main ( void )
{
 unsigned int totalcount = 0, failcount = 0;
 float grade, sum = 0;
 printf( "%s", "Enter grade (-1 to end): ");
 scanf( "%f", &grade ); // get grade
 while ( grade != -1 )
 {
 if ( ( grade >= 0 ) && ( grade <= 100 ) )
 {
 sum += grade; // sum all grades entered
 ++totalcount;
 if ( grade < 40 )
 {
 ++failcount; // count failed grades
 } // end inner if
 } // end outer if
 if ( grade != -1 )
 { // prompt to enter again
 printf("%s", "Enter grade (-1 to end): ");
 scanf( "%f", &grade ); // get grade
 } // end second if
 } // end while
 if ( totalcount == 0 )
 {
 puts("No valid grade is entered"); // invalid input
 } // end if
 else
 {
 printf("\n%u grades are entered\nAverage grade is %.2f\n%.2f students failed",
 totalcount, sum/totalcount, (float)failcount/totalcount );
 } // end else if
} // end main
```

Q3

Marking scheme

Mark	Item
2	Comment
2	Indentation & blank lines
3	Check until valid obtained
2	External function format
4	Multiplication table display
2	Display format (width of 6, right-justified)

```
#include <stdio.h>

// function prototype
void displayMultiplicationTable( int number );

// function main starts execution
int main( void )
{
   int userNumber = 0;

   // number entered by user should be within 1 - 9
   while ( ( userNumber < 1 ) || ( userNumber > 9 ) )
   {
      printf( "%s", "Select a number between 1 and 9: " );
      scanf( "%d", &userNumber );
   } // end while

   displayMultiplicationTable( userNumber );

   return 0;
} // end main


// displayMultiplicationTable
void displayMultiplicationTable( int number )
{
   unsigned int i, j;

   for ( i = 1; i <= number; ++i )
   {
      for ( j = 1; j <= i; ++j )
      {
         printf( "%6d", j * i );
      } // end inner for loop
      puts(""); // start a new line
   } // end outer for loop
} // end displayMultiplicationTable
```

Q4

Marking scheme

Mark	Item
2	Comment
2	Indentation & blank lines
2	Rand number in correct range
2	Lucky number get (loop check)
3	Array display (correct format)
4	Find lucky number and display correct message

```
// try lucky number in an array with generated random numbers

int findLuckyNumber ( int n, int a[] ); // function prototype

#include <stdio.h>
#include <time.h>
#define SIZE 100

// function main starts execution
int main( void )
{
   int array[ SIZE ];
   size_t i; // counter
   int luckyNumber = -1;
   int flagMain; // flag used in main

   /**********************************************************
   initialize the array, read in lucky number
   **********************************************************/
   srand( time( NULL ) );
   for ( i = 0; i < SIZE; ++i )
   {
      array[ i ] = 1 + rand() % 200;
   } // end for

   // prompt to enter the lucky number
   while ( ( luckyNumber < 0 ) || ( luckyNumber > 200 ) )
   { // ensure the number is 0-200
      printf("%s", "Please enter your lucky number: ");
      scanf("%d", &luckyNumber );
   } // end while

   /**********************************************************
   find lucky number in the array and show the answer
   **********************************************************/
   flagMain = findLuckyNumber( luckyNumber, array );

   if ( flagMain )
   {
      puts("Congratulations");
   } // end if
   else
   {
      puts("You are not so lucky!");
   } // end else

   /*********************************************************
   show the generated array
   *********************************************************/
   printf("\n\nThe generated array is:\n");
   for ( i = 0; i < SIZE; ++i )
   {
      printf("%5d", array[ i ] );

      if ( ( ( i + 1 ) % 10 ) == 0 )
      { // start newline after display 10 numbers on a row
         puts("");
      } // end if
   } // end for

   return 0;
} // end main

/*********************************************************
define external function to find the lucky number in the
array with random numbers
*********************************************************/
int findLuckyNumber ( int n, int a[] )
{
   int j; // counter
   int flagExternal = 0; // flag in external function

   for ( j = 0; j < SIZE; ++j )
   {
      if ( n == a[ j ] )
      { // find the lucky number and quite the loop
         flagExternal = 1;
         break;
      } // end if
   } // end for

   return flagExternal;
} // end external function
```

Q5

Marking scheme

Mark	Item
4	Comment
2	Indentation & blank lines
3	Read in name (limit <=30 characters, array size 31, %30s)
3	Read in age and check validity using loop
5	Correct info written in file in correct format
2	Ctr+z to finish entry
2	Read from user in main()
3	Correct external function format (pointers used)
4	Successfully read from file
2	Feof to check file end
4	Correct counting and calculation
2	Display in main()
4	User-friendly display

```
#include <stdio.h>
#include <stdlib.h>

void AgeStats( int *hcount, int *ocount, float *aaverage );

int main( void )
{
   char staffname[31]; // must of size 31
   int staffage;
   int headcount = 0, oldcount = 0;
   float averageage;

   FILE *fpointer;

   fpointer = fopen( "AgeStatistics.txt", "w");

   if ( fpointer == NULL )
   {
      return 1;
   } // check if file is opened successfully

   printf("Enter staff name: ");
   scanf("%30s", staffname );


   while( !feof( stdin ) ) // control+z to finish entry
   {
      printf("Enter staff age: ");
      scanf("%d", &staffage );
      while( ( staffage >80 ) || ( staffage <20 ) )
      { // check entered age range
         puts("staff age shall be within 20~80. Enter again:");
         scanf("%d", &staffage);
```

```
        } // end of inner while

        fprintf( fpointer, "%s %d\n", staffname, staffage );

        printf("Enter staff name: ");
        scanf("%30s", staffname );
    } // end of while, finish reading in from user

    fclose( fpointer );

    AgeStats( &headcount, &oldcount, &averageage );

    printf("\n\nStaff number: %d\nCount older than 60: %d\nAverage age: %.1f\n", headcount, oldcount,
        averageage);

    return 0;
} // end main


void AgeStats( int *hcount, int *ocount, float *aaverage )
{
    FILE *fpt;
    int age;
    int sum = 0;
    char nread[31];

    fpt = fopen("AgeStatistics.txt", "r");

     if ( fpt == NULL )
    {
        return 2;
    } // check if can open file successfully

    fscanf( fpt, "%s %d", nread, &age );
    while ( !feof( fpt ) )
    {
        sum += age;

        (*hcount)++;

        if ( age >= 60 )
            {(*ocount) ++;}
        fscanf( fpt, "%s %d", nread, &age );
    } // end while

    *aaverage = (float) sum / *hcount;
    fclose( fpt);
    return;
} // end external function
```