**INTERNAL MODERATION FORM**

FOR CONTINUOUS ASSESSMENTS AND EXAMINATIONS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SECTION A: COURSE AND ASSESSMENT DETAILS: *to be completed by the Course Coordinator*** | | | | | | |
| Course code | UESTC1005 | Course title | INTRODUCTORY PROGRAMMING | | | |
| Academic year | 2022/23 | SCQF level | 7 | Course credits | | 10 |
| Course leader | HASAN ABBAS | | | | | |
| Please confirm that you have reviewed the Exam Assessment and Continuous Assessment Handbooks for this task. | | | Yes | | | |
| Assessment title and number | Final Exam | | | | | |
| Assessment weighting | 60 % | | No. of markers | | 4 | |
| **Name(s) of moderator(s)** |  | | Date assessment passed to moderator | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SECTION B: MODERATOR’S REPORT on the ASSESSMENT task**   1. Comment on the overall exam layout, difficulty of the questions and the range of marks for each question. 2. Any issues or suggestions for the marking team. | | | | | |
| Is the overall quality of the assessment task appropriate? | | | Yes | | No |
| If you have answered **‘No’,** please indicate why. For example, if you disagreed with the setter’s judgement on any aspect of the assessment task. Please provide evidence and any other details: | | | | | |
| Do you recommend that any of the questions should be revised? | | | Yes | | No |
| Please indicate the recommended revisions: | | | | | |
| Any further comments: | | | | | |
| Moderator’s e-signature |  | Date completed | |  | |
| Course Coordinator Comments: The feedback given on the main exam has been taken into account and necessary changes have been made in the resit exam and solution. | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SECTION C: MODERATOR’S REPORT on the SOLUTIONS** | | | | | |
| Do you agree that the marks awarded are appropriate? | | | Yes | | No |
| If you have answered **‘No’,** please indicate why. For example, if you disagreed with the setter’s judgement on any aspect of the marks. Please provide evidence and any other details: | | | | | |
| Do you recommend that marks should be adjusted? | | | Yes | | No |
| Please indicate the recommended adjustment: | | | | | |
| Any further comments: | | | | | |
| Moderator’s e-signature |  | Date completed | |  | |
| Course Coordinator Comments: The feedback on main exam solution has also been addressed here. | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SECTION D: EXTERNAL EXAMINER’S DECISION: *to be completed by the External Examiner (if required)*** | | | | | | |
| Name of External Examiner | |  | | | | |
| Do you agree to any adjustment suggested by the Moderator? | | | | Yes | | No |
| Please indicate the rationale for your decision: | | | | | | |
| Any further comments: | | | | | | |
| e-Signature |  | | Date completed | |  | |
| ***Once completed, please upload this form electronically*** | | | | | | |

**Glasgow College UESTC**

Main Exam paper

Course name (UESTC 1005 – Introductory Programming)

Date: (remember to complete when info available from Ruoli)

Time: (remember to complete when info available from Ruoli)

Attempt all PARTS. Total 100 marks

Use one answer sheet for each of the questions in this exam.

Show all work on the answer sheet.

Make sure that your University of Glasgow and UESTC Student Identification Numbers are on all answer sheets.

An electronic calculator may be used provided that it does not allow text storage or display, or graphical display.

All graphs should be clearly labelled and sufficiently large so that all elements are easy to read.

The numbers in square brackets in the right-hand margin indicate the marks allotted to the part of the question against which the mark is shown. These marks are for guidance only.

1. (a) Write a C program that prints the perimeter of a rectangle. The user is to get two messages asking them to input the height, first, then the width. The computed perimeter must be displayed with an accuracy of 2 decimal places. [10]

The program window should show the text in Figure Q1-a:

// This C program computes the perimeter of a rectangle   
Please enter a value for the rectangle height:   
Please enter a value for the rectangle width:   
Perimeter of the Rectangle is:

**Figure Q1-a**

**Solution:**

|  |  |
| --- | --- |
|  |  |
| *#include <stdio.h>*  *int main() {*  *float rec\_width;         /\* width of the rectangle \*/*  *float rec\_height;        /\* height of the rectangle \*/*  *float rec\_perimeter;     /\* perimeter (to be computed) \*/*    *printf("\*\*\*This C program computes the perimeter of a rectangle\*\*\*\n");*  *printf("Please enter a value for the rectangle height: ");*  *scanf("%f", &rec\_height);*  *printf("Please enter a value for the rectangle width: ");*  *scanf("%f", &rec\_width);*  *rec\_perimeter = 2.0 \* (rec\_height + rec\_width); /\* perimeter = 2 \* (width + height)\*/*  *printf("Perimeter of the Rectangle is : %.2f \n", rec\_perimeter);*  *return 0;*  *}* | | |
|  | | |

Marking scheme:

2 mark on defining the variables as double or float.

2 mark on adhering to the provided display text (Figure Q1-a))

6 marks on correct code.

1. Complete the functions decimalToBinary and main(), in Figure Q1-b, to find the binary equivalent of a user input decimal number. [10]

#include <stdio.h>   
#include <math.h>   
long decimalToBinary(int decimalnum)   
{   
 long binarynum = 0;   
 int rem, temp = 1;   
 while (decimalnum!=0)   
 {   
 }   
 return binarynum;   
}   
  
int main()   
{   
 int decimalnum;   
 return 0;   
}

**Figure Q1-b**

**Solution:**

*1 #include <stdio.h>*

*2 #include <math.h>*

*3 long decimalToBinary(int decimalnum)*

*4 {*

*5    long binarynum = 0;*

*6    int rem, temp = 1;*

*7    while (decimalnum!=0)*

*8    {*

*9        rem = decimalnum%2;*

*10        decimalnum = decimalnum / 2;*

*11        binarynum = binarynum + rem\*temp;*

*12        temp = temp \* 10;*

*13    }*

*14    return binarynum;*

*15 }*

*16 int main()*

*17 {*

*18    int decimalnum;*

*19    printf("Enter a Decimal Number: ");*

*20    scanf("%d", &decimalnum);*

*21    printf("Equivalent Binary Number is: %ld", decimalToBinary(decimalnum));*

*22    return 0;*

*}*

Marking scheme:

2 marks on correct line of code to find the remainder.

2 marks on correct line of code to find the quotient.

2 marks on correct line(s) of code to find the binary value.

2 marks on correct line of code to input decimal value by user.

2 marks on correct line of code to display the binary value.

1. What is the output of the program shown in Figure Q1-c [5]

#include <stdio.h>   
int main() {   
 int number, I;   
 printf("Enter an integer: ");   
 scanf("%d", &number);   
 printf("Multiplication table of %d: \n", number);   
 for (I = 1; I <= 10; ++i) {   
 printf("%d \* %d = %d \n", number, I, number \* i);   
 }   
 return 0;   
}

**Figure Q1-c**

**Solution:**

The program generates a multiplication table. The output, if the user inputs 10, is:

*Enter an integer: 10*

*Multiplication table of 10:*

*10 \* 1 = 10*

*10 \* 2 = 20*

*10 \* 3 = 30*

*10 \* 4 = 40*

*10 \* 5 = 50*

*10 \* 6 = 60*

*10 \* 7 = 70*

*10 \* 8 = 80*

*10 \* 9 = 90*

*10 \* 10 = 100*

Marking scheme:

2 marks for printing each line on a separate line

3 marks for the correct code.

1. (a) Write a function named dec2bin that accepts a decimal number (base-10) as an input and returns the base-2 (binary number) representation. For example, dec2bin(32) should return 100000. [5]

#include <stdio.h>

#include <math.h>

*long* decimalToBinary(*int* *decimalnum*)

{

*long* binarynum = 0;

*int* rem, temp = 1;

while (*decimalnum* != 0)

{

rem = *decimalnum* % 2;

*decimalnum* = *decimalnum* / 2;

binarynum = binarynum + rem \* temp;

temp = temp \* 10;

}

return binarynum;

}

*int* main()

{

*int* decimalnum;

printf("Enter a Decimal Number: ");

scanf("%d", &decimalnum);

printf("Equivalent Binary Number is: %ld\n", decimalToBinary(decimalnum));

return 0;

}

Marking Scheme – Conversion Logic +3

Function Definition - +2

1. A palindrome is a sequence of characters that reads the same forwards and backwards. A good example is radar, as well as the number 121.

Write a program to check whether a number which is entered by the user using the keyboard is palindromic in both the base-10 (decimal) and base-2 (binary) representations. For instance, the decimal number 313 is equivalent to the binary number representation, 100111001. [10]

1. Find the sum of all numbers less than 1000 that are palindromic in both base-10 and base-2 representations. [10]

#include <stdio.h>

#define N 999

*static* *int* palindromic\_base(*int* *n*, *int* *base*);

*int* main(*void*)

{

*int* i;

*int* sum = 0;

*int* num = 0;

for (i = 1; i < N; i++)

{

if (palindromic\_base(i, 10) && palindromic\_base(i, 2))

{

printf("%d\n",i);

sum += i;

num++;

}

}

printf("There are total %d palindromes before %d\n", num, N);

printf("Total sum is %d\n", sum);

return 0;

}

*int* palindromic\_base(*int* *n*, *int* *base*)

{

*int* reversed = 0;

*const* *int* orig = *n*;

while (*n*)

{

reversed = *base* \* reversed + *n* % *base*;

*n* /= *base*;

}

return orig == reversed;

}

Marking Scheme – B Correct number conversion - +3

Palindromic logic +5

Correct code +2

* C – Evaluate sum - +6
* Evaluate condition - 4

1. (a) Given a 1-d array (not longer than 10 numbers) with both odd and even integers, write a C program that selects the odd integers from the array and sorts them in descending order. [25]

You will ask the user the lengths of the array and input the numbers. An example is as follows. User inputs are shown in bold.

Example:

Input the number of elements of the array: 5   
Input the 5 elements in the array:   
element[0]: 1   
element[1]: 2   
element[2]: 3   
element[3]: 4   
element[4]: 5   
  
The sorted array with odd number is:  
5 3 1

**Figure Q3**

You will be assessed on the following criterion:

Store the array that the user inputs [5]

Select the odd numbers to form a new array [5]

The sort of the new array [10]

Output the outcome [5]

Answer:

#include <stdio.h>

void main()

{

int arr1[10], arr2[10];

int i,j=0,k=0,l=0,n;

printf("Input the number of elements of the array:");

scanf("%d",&n);

printf("Input %d elements in the array :\n",n);

for(i=0;i<n;i++)

{

printf("element - %d : ",i);

scanf("%d",&arr1[i]);

}

for(i=0;i<n;i++)

{

if (arr1[i]%2!= 0)

{

arr2[j] = arr1[i];

j++;

}

}

for(i=0;i<j; i++)

{

for(k=0;k<j-1;k++)

{

if(arr2[k]<=arr2[k+1])

{

l=arr2[k+1];

arr2[k+1]=arr2[k];

arr2[k]=l;

}

}

}

printf("\n The sorted array with odd number is:\n");

for(i=0; i<j; i++)

{

printf("%d ", arr2[i]);

}

printf("\n\n");

}

1. (a) Write a C program to find the frequency of a specific character in the text string input by the user. The program should prompt the user to enter the string and the character to be searched. The output of the program is the number of times the target character has appeared in the text string as shown in Figure Q4-a. [10]

Enter a string: Hello friends say Hello   
Enter a character to find its frequency: H  
The frequency of the character H is: 2

**Figure Q4-a**

**Solution:**

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | *#include <stdio.h>*  **int** main() {  **char** str[1000], ch;  **int** count = 0;  printf(*"Enter a string: "*);  fgets(str, **sizeof**(str), stdin);  printf(*"Enter a character to find its frequency: "*);  scanf(*"%c"*, &ch);  **for** (**int** i = 0; str[i] != *'\0'*; ++i) {  **if** (ch == str[i])  ++count;  }  printf(*"Frequency of the character %c = %d is"*, ch, count);  **return** 0;  } |

1. The program shown in Figure Q4-b prompts the user to input a string. A user enters the following string: testdata. Based on this scenario, please answer the following:

#include <stdio.h>  
#define SIZE 80  
  
size\_t mystery2(const char \*s); // prototype  
  
int main(void)  
{  
 char string[SIZE]; // create char array  
  
 puts("Enter a string: ");  
 scanf("%79s", string);  
 printf("%d\n", mystery2(string));  
}  
  
// What does this function do?  
size\_t mystery2(const char \*s)  
{  
 size\_t x;  
  
 // loop through string  
 for (x = 0; \*s != '\0'; ++s) {  
 ++x;  
}  
  
return x;  
}

**Figure Q4-b**

1. Please describe the purpose of the mystery function in the program shown in Figure Q4-b. [5]
2. What is the output of the program shown in Figure Q4-b [5]

**Solution:**

1. The purpose of the mystery function is to determine the length of the input string.
2. Based on the user input string the program outputs the length of the string which is **8**.

(c) What is the output of the program shown in Figure Q4-c [5]

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | *#include<stdio.h>*  *#define square(x) x\*x*  **int** main()  {  **int** i;  i = 64/square(4);  printf(*"%d"*,i);  **return** 0;  } |

Figure Q4-c

**Solution:**

The program makes use of macro to define a square function. However, since there is

no parenthesis around x\*x in the macro function, the execution takes place in the following step 64/square(4) > 64/4\*4 > 16\*4 = 64. The operator ‘/’ enjoys more priority o