Indian Institute of Technology Kharagpur
Department of Computer Science and Engineering

CS10003 Programming and Data Structures

Spring Semester, 2021-2022 **Sections 3 & 4**

Long Test 1 – Part B[Marks = 50] **18-May-2022, 10:15 to 11:25**

INSTRUCTIONS

- You have 1 hour for writing and 10 minutes for submission.
- Write your answers on paper. Answers must be handwritten. Typed or written answers using an electronic device are not allowed.
- Write your name and roll number on each page. Write page numbers for each page.
- Scan all pages and collate. Create a single PDF file (of size $\leq$ 10 MB). You could also take pictures of different pages, combine them to make a single pdf file.
- The name of the file for this part should be <RollNumber>_Long1B. Ensure your roll number in the filename has only digits and uppercase characters.
- Upload your file. Make sure you click on the 'Turn in' button to submit your file.
- The said deadlines are strict after which no submissions will be allowed.
- We will not accept submission by any other means.

1. The hexadecimal number system represents numbers using 16 as the base. The numbers 0 to 15 make up the digits of this system. Symbols A,B,C,D,E,F are used to represent the values 10,11,12,13,14,15 respectively. If the hexadecimal representation of a number n is given by $a_{k-1}a_{k-2}\cdots a_1a_0$, then we have $n = \sum_{i=0}^{k-1} a_i 16^i$. For example, the hexadecimal representation of 27 is given by 1B. You could verify it by computing $1 \times 16 + 11$ which equals 27.

   Write a C program to read an integer x in usual integer format. Generate the hexadecimal representation of x in a character string s. For example, if 270 is the integer input for x then the string s must be "10E", because $270 = 16^2 \times 1 + 16 \times 0 + 1 \times 14$. Print the string s. For $x = 33$, s must be "21", and for $x = 31$, s must be "1F". 10

   Solution:

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

int main(){
  char hex[100], t;
  int x,y,d,k=0,len;

  scanf("%d", &x);
  y = x;
  if(x < 0) y = -y;
  while(y > 0){
    d = y%16;
    if(d <= 9)
      hex[k++] = '0'+ d;
```

```
    else
      hex[k++] = 65 + d - 10;
    y = y/16;
  }
  hex[k] = '\0'; len = k;
  for(k=0; k<len/2; k++){
    t = hex[k]; hex[k] = hex[len-k-1]; hex[len-k-1] = t;
  }
  if(x < 0) printf("Hexadecimal Representation: -%s\n",hex);
  else printf("Hexadecimal Representation: %s\n",hex);
}
```

2. Read a positive integer n, $n < 100$. Then read roll numbers of n students and their total marks in two integer arrays. Now, rank the roll numbers based on total in non-increasing / descending order. In case of ties in marks, they get the same rank. If k people get the same rank, say m, then the next rank starts from $m + k$. That is, if there are 3 persons who get rank 6, having received equal marks, then the rank of the person with the next highest marks is 9. At the end print the merit list with ranks, roll numbers and total marks. Do not use any library functions other than standard input and output. You can use arrays. There should be only one main function and no other functions defined by you. 10

Solution:

```
#include<stdio.h>
main()
{
int Roll [100], Marks[100], Rank[100], n, i,j,k, temp1, temp2, count, merit;
scanf("%d", &n);
printf("n = %d\n", n);
for (k=0; k<n;k++) scanf("%d%d", &Roll[k], &Marks[k]);
printf(" Input: \n");
for (k=0;k<n;k++) printf("Roll [%d] = %d, Marks = %d \n", k, Roll[k], Marks[k]);
printf("\n");
for(i=0; i< n-1; i++){
for (j=0; j<n-i-1;j++) {
    if(Marks[j] < Marks[j+1]) {
        temp1 = Roll[j];
        Roll[j] = Roll[j+1];
        Roll[j+1] = temp1 ;
        temp2 = Marks[j];
        Marks[j] = Marks[j+1];
        Marks[j+1] = temp2 ;
    }
    }

}
merit = 1;
Rank[0] = 1;
count = 1;
for(k = 0; k < n-1; k++)
{
    if (Marks[k] == Marks [k+1])
    {
        Rank[k+1] = Rank[k];
        count++;
    }
    else

    {
```

```
        Rank[k+1] = Rank[k] + count;
        count = 1;
    }
}
printf("Merit List \n: ");
printf("Rank  Roll_Number  Marks \n");
for (k=0;k<n;k++) printf("%5d %10d %7d \n", Rank[k], Roll[k], Marks[k]);
printf("\n");
}
```

3. It is known that a number is divisible by 9 if sum of its digits is divisible by 9. For example, if $n = 9432$, the sum of digits of $n = 9+4+3+2 = 18$. This sum is divisible by 9, and therefore the number n is divisible by 9. Write a function, $f(n)$, that finds the sum of the digits of n and determines whether the sum is divisible by 9. If so, it returns 1, else it returns 0. [10]

Solution:

```
int f(int n){
  int dsum=0, t;
  t = n;
  while (t != 0){
    dsum += t%10;
    t /= 10;
  }
  return (dsum%9 == 0)?1:0;
}
```

4. A number p is called a Sophie Germain prime (named after the French mathematician Sophie Germain) if both p and $2p+1$ are prime numbers. Write a C function $is\_SGprime(int\ a, int\ b)$ that on input two positive integers a, b with $a < b$ lists all Sophie Germain primes in the interval $[a, b]$. You are not allowed to use any library functions other than $sqrt()$ and standard input/output. There is no need to write $main()$ function. Only write the function definition. For example the output of $is\_SGprime(250, 500)$ should be

```
Sophie Germain primes in the interval [250 500] are
251 281 293 359 419 431 443 491
```

[10]

Solution:

```
void is_SGprime(int a, int b){
  int num, i, k, flag1, flag2;
  printf("Sophie Germain primes in the interval [%d,%d] are\n",a,b);

  for(num=a; num<=b; num++){
    flag1 = flag2 = 1;
    if (num == 1)
      flag1 = 0;
    if (num%2 == 0)
      flag1 = 0;
    for(i=3; i*i<=num; i++){
      if (num%i == 0)
        flag1 = 0;
    }
    k = 2*num+1;
    if (k == 1)
```

```
      flag1 = 0;
    if (k%2 == 0)
      flag1 = 0;
    for(i=3; i*i<=k; i++){
      if (k%i == 0)
        flag2 = 0;
    }

    if (flag1 && flag2)
      printf("%d ", num);
  }
  printf("\n\n");
}
```

5. Consider the following program segment. Write the output of the program. Also write down the number of times the function `fn` is called.

```
#include <stdio.h>
int fn(int m, int n)
{
    if(m==n)
        return(m);
    if(n==0)
        return (1);
    else if(n==1)
        return (m);
    else
        return(fn(m-1, n)+fn(m-1, n-1));
}

void main()
{
    printf("value=%d \n",   fn(5, 3));
}
```

10

Solution: 14