WEEK 12[SESSION 1]

Question **1**

Correct

Marked out of 1.00

Flag question

A binary number is a combination of 1s and 0s. Its n^{th} least significant digit is the n^{th} digit starting from the right starting with 1. Given a decimal number, convert it to binary and determine the value of the the 4^{th} least significant digit.

```c
int fourthBit(int number)
{
    int arr[100],e=0;
    while(number>0){
        int m=number%2;
        arr[e]=m;
        e++;
        number/=2;
    }
    int start=0;
    int last =e;
    while(start<last){
        int temp=arr[start];
        arr[start]=arr[last];
        arr[last]=temp;
        start++;
        last--;
    }
    return arr[e-3];
}
```

	Test	Expected	Got	
✓	printf("%d", fourthBit(32))	0	0	✓
✓	printf("%d", fourthBit(77))	1	1	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

Flag question

Determine the factors of a number (i.e., all positive integer values that evenly divide into a number) and then return the p^{th} element of the list, sorted ascending. If there is no p^{th} element, return 0.

Example

n = 20

p = 3

The factors of 20 in ascending order are {1, 2, 4, 5, 10, 20}. Using 1-based indexing, if p = 3, then 4 is returned. If p > 6, 0 would be returned.

```
9
10  long pthFactor(long n, long p)
11  {
12      int arr[100],e=0;
13      for(int i=1;i<=n;i++){
14          if(n%i==0){
15          arr[e]=i;
16          e++;
17        }
18      }
19      if(p<=e){
20          return arr[p-1];
21      }else{
22          return 0;
23      }
24  }
```

	Test	Expected	Got	
✓	printf("%ld", pthFactor(10, 3))	5	5	✓
✓	printf("%ld", pthFactor(10, 5))	0	0	✓
✓	printf("%ld", pthFactor(1, 1))	1	1	✓

Passed all tests! ✓

WEEK 12[SESSION 1]

Question **1**

Correct

Marked out of 1.00

Flag question

You are a bank account hacker. Initially you have 1 rupee in your account, and you want exactly ***N*** rupees in your account. You wrote two hacks, first hack can multiply the amount of money you own by 10, while the second can multiply it by 20. These hacks can be used any number of time. Can you achieve the desired amount ***N*** using these hacks.

```
int myFunc(int n)
{
    if(n==1)
        return 1;
     if(n%10==0)
        if(myFunc(n/10)==1)
            return 1;

    if(n%20==0)
        if(myFunc(n/20)==1)
            return 1;


    return 0;
}
```

	Test	Expected	Got	
✓	printf("%d", myFunc(1))	1	1	✓
✓	printf("%d", myFunc(2))	0	0	✓
✓	printf("%d", myFunc(10))	1	1	✓
✓	printf("%d", myFunc(25))	0	0	✓
✓	printf("%d", myFunc(200))	1	1	✓

Passed all tests! ✓

Question **2**

Correct

Marked out of 1.00

Flag question

Find the number of ways that a given integer, ***X***, can be expressed as the sum of the ***Nth*** powers of unique, natural numbers.

For example, if ***X*** = ***13*** and ***N*** = ***2***, we have to find all combinations of unique squares adding up to ***13***. The only solution is $2^2 + 3^2$.

```c
int powerSum(int x, int m, int n)
{
    int temp=1;
    for(int i=1;i<=n;i++){
        temp*=m;
    }
    if(temp==x){
        return 1;
    }
    if(temp>x){
        return 0;
    }
    return powerSum(x,m+1,n)+powerSum(x-temp,m+1,n);
}
```

	Test	Expected	Got	
✓	`printf("%d", powerSum(10, 1, 2))`	1	1	✓

Passed all tests! ✓