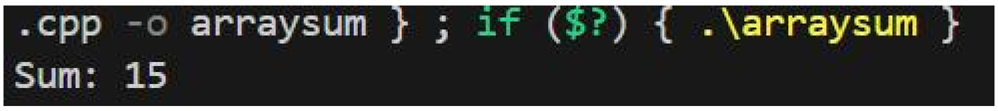
NAME:Lakshay Verma UID: 22BCS15481
CLASS: 620/B

DAY - 3

Question 1. Array Sum

```
#include <iostream>
#include <numeric>
using namespace std;
int arraySum(int arr[], int size) {
   return accumulate(arr, arr + size, 0);
}

int main() {
   int arr[] = {1, 2, 3, 4, 5};
   int size = sizeof(arr) / sizeof(arr[0]);
   cout << "Sum: " << arraySum(arr, size) << std::endl;
   return 0;
}
```

OUTPUT:

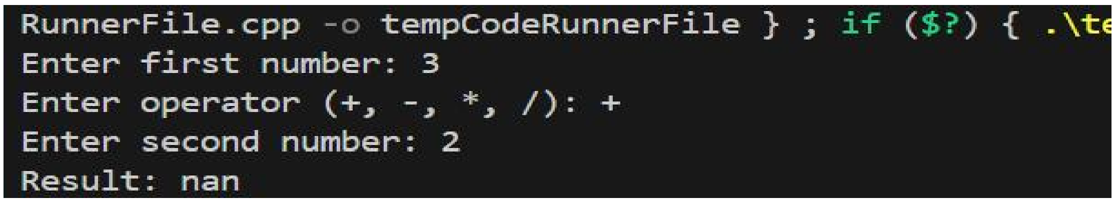


```
.cpp -o arraysum } ; if ($?) { .\arraysum }
Sum: 15
```

Question 2. Calculator

```
#include <iostream>
using namespace std;
double calculator(double num1, double num2, char op) {
   double result;
   switch (op) {
      case '+':
         result = num1 + num2;
         break;
      case '-':
         result = num1 - num2;
         break;
      case '*':
         result = num1 * num2;
         break;
      case '/':
         if (num2 != 0) {
            result = num1 / num2;
         } else {
            cerr << "Error: Division by zero." << endl;
            result = 0;
         }
         break;
      default:
         cerr << "Error: Invalid operator." << endl;
         result = 0;

   return result;
   }
}
int main() {
   double num1, num2;
   char op;
   cout << "Enter first number: ";
   cin >> num1;
   cout << "Enter operator (+, -, *, /): ";
   cin >> op;
   cout << "Enter second number: ";
   cin >> num2;

   double result = calculator(num1, num2, op);
   cout << "Result: " << result << endl;

   return 0;
}
```

OUTPUT

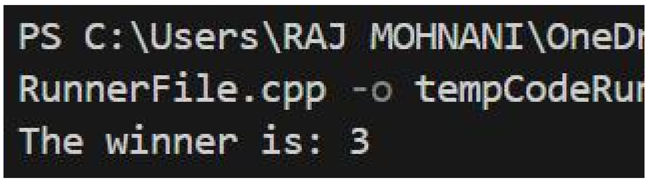


```
RunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\te
Enter first number: 3
Enter operator (+, -, *, /): +
Enter second number: 2
Result: nan
```

Question 3.

```
#include <iostream>
using namespace std;

int findTheWinnerUtil(int n, int k) {
    if (n == 1)
        return 0; // The first position is the winner
    else
        return (findTheWinnerUtil(n - 1, k) + k) % n;
}

int findTheWinner(int n, int k) {
    return findTheWinnerUtil(n, k) + 1; // To convert 0-based index to 1-based
}

int main() {
    int n = 5, k = 2;
    cout << "The winner is: " << findTheWinner(n, k) << endl;
    return 0;
}
```

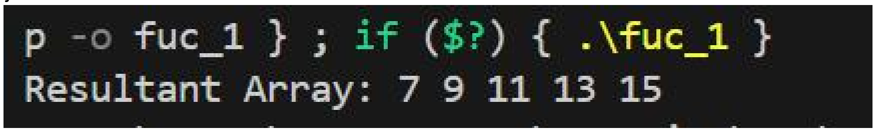
OUTPUT:



```
PS C:\Users\RAJ MOHNANI\OneDr
RunnerFile.cpp -o tempCodeRur
The winner is: 3
```

Question 4.
Two array Sum

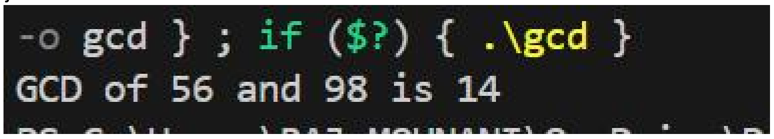
```
#include <iostream>
#include <algorithm>

void arraySum(const int arr1[], const int arr2[], int result[], int size) {
  std::transform(arr1, arr1 + size, arr2, result, std::plus<int>());
}

int main() {
  const int size = 5;
  int arr1[size] = {1, 2, 3, 4, 5};
  int arr2[size] = {6, 7, 8, 9, 10};
  int result[size];

  arraySum(arr1, arr2, result, size);

  std::cout << "Resultant Array: ";
  for(int i = 0; i < size; ++i) {
    std::cout << result[i] << " ";
  }
  std::cout << std::endl;

  return 0;
}
```



```
p -o fuc_1 } ; if ($?) { .\fuc_1 }
Resultant Array: 7 9 11 13 15
```

Question 5. GCD

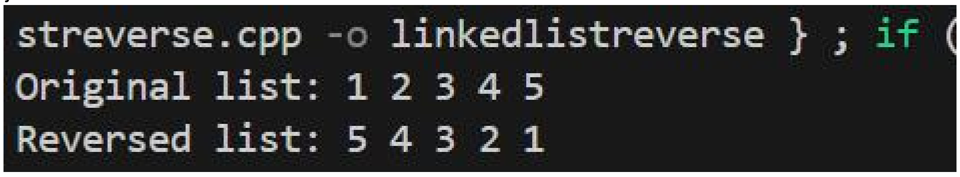
```
#include <iostream>

int gcd(int a, int b) {
  while (b != 0) {
    int temp = b;
    b = a % b;
    a = temp;
  }
  return a;
}

int main() {
  int a = 56;
  int b = 98;
  std::cout << "GCD of " << a << " and " << b << " is " << gcd(a, b) << std::endl;
  return 0;
}
```



```
-o gcd } ; if ($?) { .\gcd }
GCD of 56 and 98 is 14
```

Question 6. linkedlist revese

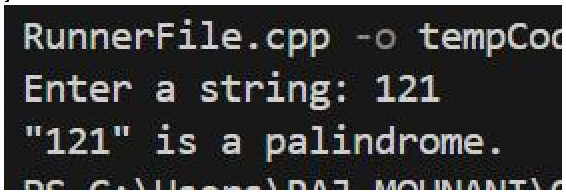
```
#include <iostream>

struct ListNode {
    int val;
    ListNode* next;
    ListNode(int x) : val(x), next(nullptr) {}
};

ListNode* reverseList(ListNode* head) {
    ListNode* prev = nullptr;
    ListNode* current = head;
    ListNode* next = nullptr;

    while (current != nullptr) {

        next = current->next;

        current->next = prev;

        prev = current;
        current = next;
    }

    return prev;
}

void printList(ListNode* head) {
    ListNode* current = head;
    while (current != nullptr) {
        std::cout << current->val << " ";
        current = current->next;
    }
    std::cout << std::endl;
}

int main() {

    ListNode* head = new ListNode(1);
    head->next = new ListNode(2);
    head->next->next = new ListNode(3);
    head->next->next->next = new ListNode(4);
    head->next->next->next->next = new ListNode(5);

    std::cout << "Original list: ";
    printList(head);

    head = reverseList(head);

    std::cout << "Reversed list: ";
    printList(head);


    ListNode* current = head;
    while (current != nullptr) {
        ListNode* next = current->next;
        delete current;
        current = next;
    }

    return 0;
}
```



```
streverse.cpp -o linkedlistreverse } ; if (
Original list: 1 2 3 4 5
Reversed list: 5 4 3 2 1
```

Question 7. Palindrome using function

```
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;

bool isPalindrome(const std::string& str) {
    std::string cleanedStr;
    for (char ch : str) {
        if (isalnum(ch)) {
            cleanedStr += tolower(ch);
        }
    }

    string reversedStr = cleanedStr;
    reverse(reversedStr.begin(), reversedStr.end());

    return cleanedStr == reversedStr;
}

int main() {
    string input;
    cout << "Enter a string: ";
    getline(cin, input);

    if (isPalindrome(input)) {
        cout << "\"" << input << "\" is a palindrome." << endl;
    } else {
        cout << "\"" << input << "\" is not a palindrome." << endl;
    }

    return 0;
}
```

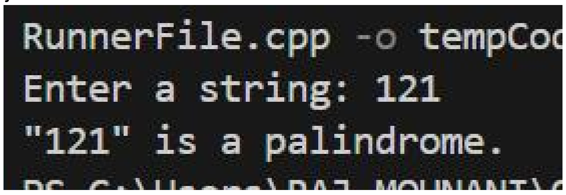


```
RunnerFile.cpp -o tempCod
Enter a string: 121
"121" is a palindrome.
PS C:\Users\RAJ MOHNANI\C
```

Question 8. Palindrome using linkedlist

```
#include <iostream>
```

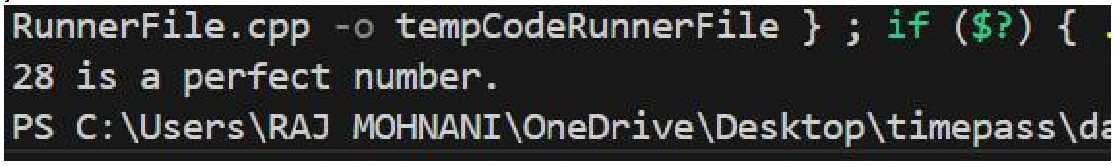
```
using namespace std;

struct ListNode {
    int val;
    ListNode *next;
    ListNode(int x) : val(x), next(NULL) {}
};

bool isPalindromeUtil(ListNode **left, ListNode *right) {
    // Base case: when right becomes NULL
    if (right == NULL)
        return true;

    // Recursively check the next nodes
    bool isp = isPalindromeUtil(left, right->next);
    if (!isp)
        return false;

    // Check values from left and right
    bool isp1 = (right->val == (*left)->val);

    // Move left pointer one step to the right
    *left = (*left)->next;

    return isp1;
}

bool isPalindrome(ListNode *head) {
    return isPalindromeUtil(&head, head);
}

int main() {
    ListNode *head = new ListNode(1);
    head->next = new ListNode(2);
    head->next->next = new ListNode(2);
    head->next->next->next = new ListNode(1);

    if (isPalindrome(head))
        cout << "Linked List is a palindrome" << endl;
    else
        cout << "Linked List is not a palindrome" << endl;

    return 0;
}
```



```
RunnerFile.cpp -o tempCoc
Enter a string: 121
"121" is a palindrome.
```

Question 9. Perfect number

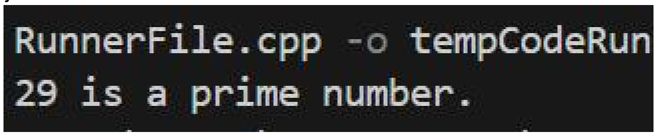
```
#include <iostream>

bool isPerfectNumber(int num) {
    if (num <= 1) return false;
    int sum = 0;

    for (int i = 1; i <= num / 2; ++i) {
        if (num % i == 0) {
            sum += i;
        }
    }

    return sum == num;
}

int main() {
    int num = 28;
    if (isPerfectNumber(num)) {
        std::cout << num << " is a perfect number." << std::endl;
    } else {
        std::cout << num << " is not a perfect number." << std::endl;
    }
    return 0;
}
```



```
RunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) {
28 is a perfect number.
PS C:\Users\RAJ MOHNANI\OneDrive\Desktop\timepass\d
```

Question 10. Prime number

```
#include <iostream>
#include <cmath>

bool isPrime(int n) {
    if (n <= 1) return false;
    for (int i = 2; i <= std::sqrt(n); ++i) {
        if (n % i == 0) return false;
    }
    return true;
}

int main() {
    int num = 29;
    if (isPrime(num)) {
        std::cout << num << " is a prime number." << std::endl;
    } else {
        std::cout << num << " is not a prime number." << std::endl;
    }
    return 0;
}
```

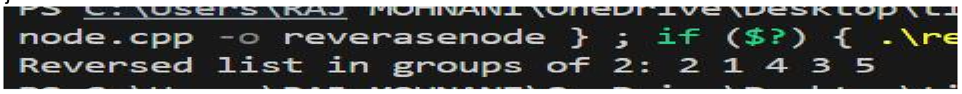


```
RunnerFile.cpp -o tempCodeRun
29 is a prime number.
```

Question 11. reversal

```
#include <iostream>
using namespace std;
```

```
struct ListNode {
    int val;
    ListNode *next;
    ListNode(int x) : val(x), next(NULL) {}
};

ListNode* reverseKGroup(ListNode* head, int k) {
    ListNode* curr = head;
    int count = 0;


    while (curr && count < k) {
        curr = curr->next;
        count++;
    }

    if (count == k) {

        curr = reverseKGroup(curr, k);
        while (count > 0) {
            ListNode* temp = head->next;
            head->next = curr;
            curr = head;
            head = temp;
            count--;
        }
        head = curr;
    }
    return head;
}

void printList(ListNode* head) {
    while (head != NULL) {
        cout << head->val << " ";
        head = head->next;
    }
    cout << endl;
}

int main() {
    ListNode* head = new ListNode(1);
    head->next = new ListNode(2);
    head->next->next = new ListNode(3);
    head->next->next->next = new ListNode(4);
    head->next->next->next->next = new ListNode(5);

    int k = 2;
    head = reverseKGroup(head, k);
    cout << "Reversed list in groups of " << k << ": ";
    printList(head);

    return 0;
}
```



```
PS C:\Users\RAJ MOHNANI\OneDrive\Desktop\
node.cpp -o reverasenode } ; if ($?) { .\re
Reversed list in groups of 2: 2 1 4 3 5
```