C++ PRACTICE PROGRAMS

BY

S.AIME

1.Hello world program

```cpp
#include <iostream>

int main() {
    std::cout << "Hello World!";
    return 0;
}
```

2. check prime number

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

int main() {
    int i, n;
    bool isPrime = true;

    cout << "Enter a positive integer: ";
    cin >> n;

    // 0 and 1 are not prime numbers
    if (n == 0 || n == 1) {
        isPrime = false;
    }
    else {
        for (i = 2; i <= n / 2; ++i) {
            if (n % i == 0) {
                isPrime = false;
                break;
            }
        }
    }
    if (isPrime)
        cout << n << " is a prime number";
    else
        cout << n << " is not a prime number";
```

```
    return 0;
```

Pyramids and patterners

*Example 1: Program to print half pyramid using **

```
*
* *
* * *
* * * *
* * * * *
```

Source Code

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

int main()
{
    int rows;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = 1; i <= rows; ++i)
    {
        for(int j = 1; j <= i; ++j)
        {
            cout << "* ";
        }
        cout << "\n";
    }
    return 0;
}
```

Example 2: Program to print half pyramid a using numbers

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Source Code

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

int main()
{
    int rows;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = 1; i <= rows; ++i)
    {
        for(int j = 1; j <= i; ++j)
        {
            cout << j << " ";
        }
        cout << "\n";
    }
    return 0;
}
```

Example 3: Program to print half pyramid using alphabets

```
A
B B
C C C
D D D D
E E E E E
```

Source Code

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

int main()
{
    char input, alphabet = 'A';

    cout << "Enter the uppercase character you want to print in the last row: ";
    cin >> input;

    for(int i = 1; i <= (input-'A'+1); ++i)
    {
        for(int j = 1; j <= i; ++j)
        {
            cout << alphabet << " ";
        }
        ++alphabet;

        cout << endl;
    }
    return 0;
}
```

Programs to print inverted half pyramid using * and numbers

---

*Example 4: Inverted half pyramid using **

```
* * * * *
* * * *
* * *
* *
*
```

Source Code

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

int main()
{
    int rows;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = rows; i >= 1; --i)
    {
        for(int j = 1; j <= i; ++j)
        {
            cout << "* ";
        }
        cout << endl;
    }

    return 0;
}
```

Example 5: Inverted half pyramid using numbers

```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

Source Code

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

int main()
{
    int rows;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = rows; i >= 1; --i)
    {
        for(int j = 1; j <= i; ++j)
        {
            cout << j << " ";
        }
        cout << endl;
    }

    return 0;
}
```

Programs to display pyramid and inverted pyramid using * and digits

---

*Example 6: Program to print full pyramid using **

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
```

Source Code

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

int main()
{
    int space, rows;

    cout <<"Enter number of rows: ";
    cin >> rows;

    for(int i = 1, k = 0; i <= rows; ++i, k = 0)
    {
        for(space = 1; space <= rows-i; ++space)
        {
            cout <<"  ";
        }

        while(k != 2*i-1)
        {
            cout << "* ";
            ++k;
        }
        cout << endl;
    }
    return 0;
}
```

Example 7: Program to print pyramid using numbers

```
        1
      2 3 2
    3 4 5 4 3
  4 5 6 7 6 5 4
5 6 7 8 9 8 7 6 5
```

Source Code

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

int main()
{
    int rows, count = 0, count1 = 0, k = 0;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = 1; i <= rows; ++i)
    {
        for(int space = 1; space <= rows-i; ++space)
        {
            cout << "  ";
            ++count;
        }

        while(k != 2*i-1)
        {
            if (count <= rows-1)
            {
                cout << i+k << " ";
                ++count;
            }
            else
            {
                ++count1;
                cout << i+k-2*count1 << " ";
            }
```

```
            ++k;
        }
        count1 = count = k = 0;

        cout << endl;
    }
    return 0;
}
```

---

*Example 8: Inverted full pyramid using **

```
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

Source Code

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

int main()
{
    int rows;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = rows; i >= 1; --i)
    {
        for(int space = 0; space < rows-i; ++space)
            cout << "  ";

        for(int j = i; j <= 2*i-1; ++j)
            cout << "* ";
```

```
        for(int j = 0; j < i-1; ++j)
            cout << "* ";

        cout << endl;
    }

    return 0;
}
```

Example 9: Print Pascal's triangle

```
          1
        1   1
      1   2   1
    1   3   3    1
  1  4    6   4   1
 1  5   10   10  5   1
```

Source Code

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

int main()
{
    int rows, coef = 1;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = 0; i < rows; i++)
    {
        for(int space = 1; space <= rows-i; space++)
            cout <<"  ";

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

```
        {
            if (j == 0 || i == 0)
                coef = 1;
            else
                coef = coef*(i-j+1)/j;

            cout << coef << "   ";
        }
        cout << endl;
    }

    return 0;
}
```

Example 10: Print Floyd's Triangle.

```
1
2 3
4 5 6
7 8 9 10
```

Source Code

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

int main()
{
    int rows, number = 1;

    cout << "Enter number of rows: ";
    cin >> rows;

    for(int i = 1; i <= rows; i++)
    {
        for(int j = 1; j <= i; ++j)
        {
```

```
            cout << number << " ";
            ++number;
        }

        cout << endl;
    }

    return 0;
}
```

Example: Compute quotient and remainder

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

int main()
{
    int divisor, dividend, quotient, remainder;

    cout << "Enter dividend: ";
    cin >> dividend;

    cout << "Enter divisor: ";
    cin >> divisor;

    quotient = dividend / divisor;
    remainder = dividend % divisor;

    cout << "Quotient = " << quotient << endl;
    cout << "Remainder = " << remainder;

    return 0;
}
```

Example 1: Check Whether Number is Even or Odd using if else

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

int main()
{
    int n;

    cout << "Enter an integer: ";
    cin >> n;

    if ( n % 2 == 0)
        cout << n << " is even.";
    else
        cout << n << " is odd.";

    return 0;
}
```

Example: Check Vowel or a Consonant Manually

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

int main() {
    char c;
    int isLowercaseVowel, isUppercaseVowel;

    cout << "Enter an alphabet: ";
    cin >> c;

    // evaluates to 1 (true) if c is a lowercase vowel
    isLowercaseVowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c ==
'u');

    // evaluates to 1 (true) if c is an uppercase vowel
    isUppercaseVowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c ==
'U');

    // show error message if c is not an alphabet
    if (!isalpha(c))
```

```
      printf("Error! Non-alphabetic character.");
    else if (isLowercaseVowel || isUppercaseVowel)
        cout << c << " is a vowel.";
    else
        cout << c << " is a consonant.";

    return 0;
}
```

Example 1: Find Largest Number Using if Statement

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

int main() {
    float n1, n2, n3;

    cout << "Enter three numbers: ";
    cin >> n1 >> n2 >> n3;

    if(n1 >= n2 && n1 >= n3)
        cout << "Largest number: " << n1;

    if(n2 >= n1 && n2 >= n3)
        cout << "Largest number: " << n2;

    if(n3 >= n1 && n3 >= n2)
        cout << "Largest number: " << n3;

    return 0;
}
```

Example: Roots of a Quadratic Equation

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

```
int main() {

    float a, b, c, x1, x2, discriminant, realPart, imaginaryPart;
    cout << "Enter coefficients a, b and c: ";
    cin >> a >> b >> c;
    discriminant = b*b - 4*a*c;

    if (discriminant > 0) {
        x1 = (-b + sqrt(discriminant)) / (2*a);
        x2 = (-b - sqrt(discriminant)) / (2*a);
        cout << "Roots are real and different." << endl;
        cout << "x1 = " << x1 << endl;
        cout << "x2 = " << x2 << endl;
    }

    else if (discriminant == 0) {
        cout << "Roots are real and same." << endl;
        x1 = -b/(2*a);
        cout << "x1 = x2 =" << x1 << endl;
    }

    else {
        realPart = -b/(2*a);
        imaginaryPart =sqrt(-discriminant)/(2*a);
        cout << "Roots are complex and different."  << endl;
        cout << "x1 = " << realPart << "+" << imaginaryPart << "i" << endl;
        cout << "x2 = " << realPart << "-" << imaginaryPart << "i" << endl;
    }

    return 0;
}
```

Example 1: Fibonacci Series up to n number of terms

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

int main() {
    int n, t1 = 0, t2 = 1, nextTerm = 0;

    cout << "Enter the number of terms: ";
```

```
    cin >> n;

    cout << "Fibonacci Series: ";

    for (int i = 1; i <= n; ++i) {
        // Prints the first two terms.
        if(i == 1) {
            cout << t1 << ", ";
            continue;
        }
        if(i == 2) {
            cout << t2 << ", ";
            continue;
        }
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;

        cout << nextTerm << ", ";
    }
    return 0;
}
```

Example 1: Find GCD using while loop

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

int main()
{
    int n1, n2;

    cout << "Enter two numbers: ";
    cin >> n1 >> n2;

    while(n1 != n2)
    {
        if(n1 > n2)
            n1 -= n2;
        else
```

```
            n2 -= n1;
    }

    cout << "HCF = " << n1;
    return 0;
}
```

Example 1: Find LCM

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

int main()
{
    int n1, n2, max;

    cout << "Enter two numbers: ";
    cin >> n1 >> n2;

    // maximum value between n1 and n2 is stored in max
    max = (n1 > n2) ? n1 : n2;

    do
    {
        if (max % n1 == 0 && max % n2 == 0)
        {
            cout << "LCM = " << max;
            break;
        }
        else
            ++max;
    } while (true);

    return 0;
}
```

Example: C++ Program to Reverse an Integer

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

int main() {
    int n, reversedNumber = 0, remainder;

    cout << "Enter an integer: ";
    cin >> n;

    while(n != 0) {
        remainder = n%10;
        reversedNumber = reversedNumber*10 + remainder;
        n /= 10;
    }

    cout << "Reversed Number = " << reversedNumber;

    return 0;
}
```

Example 1: Compute Power Manually

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

int main()
{
    int exponent;
    float base, result = 1;

    cout << "Enter base and exponent respectively:  ";
    cin >> base >> exponent;
```

```
    cout << base << "^" << exponent << " = ";

    while (exponent != 0) {
        result *= base;
        --exponent;
    }

    cout << result;

    return 0;
}
```

Example: Check Prime Number

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

int main() {
    int i, n;
    bool isPrime = true;

    cout << "Enter a positive integer: ";
    cin >> n;

    // 0 and 1 are not prime numbers
    if (n == 0 || n == 1) {
        isPrime = false;
    }
```

```
    else {
        for (i = 2; i <= n / 2; ++i) {
            if (n % i == 0) {
                isPrime = false;
                break;
            }
        }
    }
    if (isPrime)
        cout << n << " is a prime number";
    else
        cout << n << " is not a prime number";

    return 0;
}
```

Example: Check Armstrong Number of 3 Digits

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

int main() {
    int num, originalNum, remainder, result = 0;
    cout << "Enter a three-digit integer: ";
    cin >> num;
    originalNum = num;

    while (originalNum != 0) {
        // remainder contains the last digit
        remainder = originalNum % 10;
```

```
        result += remainder * remainder * remainder;

        // removing last digit from the orignal number
        originalNum /= 10;
    }

    if (result == num)
        cout << num << " is an Armstrong number.";
    else
        cout << num << " is not an Armstrong number.";

    return 0;
}
```

Example: Decimal to binary

```
#include <iostream>

using namespace std;

int main()
{
int a[100],b,n,i=0,j;

cout<<"enter decimal value";
cin>>b;
```

```
while(b!=0){
    a[i++]=b%2;
    b=b/2;
}
cout<<"the binary is: ";
for(j=i-1;j>=0;j--){
    cout<<a[j];
}

    return 0;
}
```

Sorting algorithm

Insertion sort

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

// Function to print an array
void printArray(int array[], int size) {
  for (int i = 0; i < size; i++) {
    cout << array[i] << " ";
  }
  cout << endl;
}

void insertionSort(int array[], int size) {
  for (int step = 1; step < size; step++) {
    int key = array[step];
    int j = step - 1;

    // Compare key with each element on the left of it until an element smaller
than
    // it is found.
    // For descending order, change key<array[j] to key>array[j].
    while (key < array[j] && j >= 0) {
      array[j + 1] = array[j];
      --j;
    }
    array[j + 1] = key;
```

```
  }
}

// Driver code
int main() {
  int data[] = {9, 5, 1, 4, 3};
  int size = sizeof(data) / sizeof(data[0]);
  insertionSort(data, size);
  cout << "Sorted array in ascending order:\n";
  printArray(data, size);
}
```

Bubble sort

```
// Bubble sort in C++

#include <iostream>
using namespace std;

// function to perform bubble sort
void bubbleSort(int array[], int size) {

  // loop to access each array element
  for (int step = 0; step < (size-1); ++step) {

    // loop to compare array elements
    for (int i = 0; i < size - (step-1); ++i) {

      // compare two adjacent elements
      // change > to < to sort in descending order
      if (array[i] > array[i + 1]) {

        // swapping occurs if
        // the first element is greater than second
        int temp = array[i];
        array[i] = array[i + 1];
        array[i + 1] = temp;
      }
    }
  }
}
```

```
// function to print array
void printArray(int array[], int size) {
  for (int i = 0; i < size; ++i) {
    cout << "  " << array[i];
  }
  cout << "\n";
}

// main function
int main() {
  int data[] = {-2, 45, 0, 11, -9};

  // find the length of the array
  int size = sizeof(data) / sizeof(data[0]);

  bubbleSort(data, size);

  cout << "Sorted Array in Ascending Order:\n";

  // call the function to print the array
  printArray(data, size);
}
```

Quick sort

```
// Quick sort in C++

#include <iostream>
using namespace std;

// function to swap elements
void swap(int *a, int *b) {
  int t = *a;
  *a = *b;
  *b = t;
}

// function to print the array
void printArray(int array[], int size) {
  int i;
```

```
  for (i = 0; i < size; i++)
    cout << array[i] << " ";
  cout << endl;
}

// function to rearrange array (find the partition point)
int partition(int array[], int low, int high) {

  // select the rightmost element as pivot
  int pivot = array[high];

  // pointer for greater element
  int i = (low - 1);

  // traverse each element of the array
  // compare them with the pivot
  for (int j = low; j < high; j++) {
    if (array[j] <= pivot) {

      // if element smaller than pivot is found
      // swap it with the greater element pointed by i
      i++;

      // swap element at i with element at j
      swap(&array[i], &array[j]);
    }
  }

  // swap pivot with the greater element at i
  swap(&array[i + 1], &array[high]);

  // return the partition point
  return (i + 1);
}

void quickSort(int array[], int low, int high) {
  if (low < high) {

    // find the pivot element such that
    // elements smaller than pivot are on left of pivot
    // elements greater than pivot are on righ of pivot
    int pi = partition(array, low, high);
```

```
    // recursive call on the left of pivot
    quickSort(array, low, pi - 1);

    // recursive call on the right of pivot
    quickSort(array, pi + 1, high);
  }
}

// Driver code
int main() {
  int data[] = {8, 7, 6, 1, 0, 9, 2};
  int n = sizeof(data) / sizeof(data[0]);

  cout << "Unsorted Array: \n";
  printArray(data, n);

  // perform quicksort on data
  quickSort(data, 0, n - 1);

  cout << "Sorted array in ascending order: \n";
  printArray(data, n);
}
```

2.

```
#include<iostream>

using namespace std;

int partition(int arr[], int low, int high)

{

int temp;

int pivot = arr[high]; // assuming last element of the array as the pivot element
```

```
int i = (low - 1); // assuming the index of i pointer as one position less than the first element

for (int j = low; j <= high - 1; j++) // assuming the index of j pointer as the first position

{



// If current element is smaller than or equal to pivot

if (arr[j] <= pivot)

{

i++; // increment index of i pointer and swap the elemets at index i and j

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}



// swapping the pivot (last) element and element at i + 1 index

temp = arr[i + 1];
```

```
arr[i + 1] = arr[high];

arr[high] = temp;



// returning the index of pivot element having lesser elements to the left and greater elements to

the right

return (i + 1);

}

void quick_sort(int arr[], int low, int high)

{

if (low < high)

{



// partitioning the single array into two sub-arrays

int pi = partition(arr, low, high);



// sorting the sub-arrays
```

```
quick_sort(arr, low, pi - 1);

quick_sort(arr, pi + 1, high);

}

}

int print(int arr[], int n)

{

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

{

cout << arr[i] << " ";

}

}



int main()

{

int n, i;

cin >> n;
```

```
int arr[n];

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

{

cin >> arr[i];

}

quick_sort(arr, 0, n - 1);

print(arr, n);

}
```

Rotating donut

```
k;double sin()
         ,cos();main(){float A=
       0,B=0,i,j,z[1760];char b[
     1760];printf("\x1b[2J");for(;;
  ){memset(b,32,1760);memset(z,0,7040)
  ;for(j=0;6.28>j;j+=0.07)for(i=0;6.28
```

```
 >i;i+=0.02){float c=sin(i),d=cos(j),e=
 sin(A),f=sin(j),g=cos(A),h=d+2,D=1/(c*
 h*e+f*g+5),l=cos      (i),m=cos(B),n=s\
in(B),t=c*h*g-f*        e;int x=40+30*D*
(l*h*m-t*n),y=            12+15*D*(l*h*n
+t*m),o=x+80*y,          N=8*((f*e-c*d*g
 )*m-c*d*e-f*g-l        *d*n);if(22>y&&
 y>0&&x>0&&80>x&&D>z[o]){z[o]=D;;;b[o]=
 ".,-~:;=!*#$@"[N>0?N:0];}}/*#****!!-*/
  printf("\x1b[H");for(k=0;1761>k;k++)
   putchar(k%80?b[k]:10);A+=0.04;B+=
     0.02;}}/*****###*******!!=;:~
       ~::==!!!**********!!!==::-
         .,~~;;;========;;;:~-.
             ..,--------,*/
```