

Rekursif

Struktur Data dan Algoritma

“Tugas Rekursif II”



Disusun Oleh :

Hera Karmila (0110217085)

Muhammad Azhar Rasyad (0110217029)

Teknik Informatika 1
Sekolah Tinggi Teknologi Terpadu Nurul Fikri
2018

Soal



Soal Latihan

- Buatlah program untuk menyelesaikan masalah berikut:
 1. Membentuk barisan bilangan Fibonacci.
 2. Menghitung nilai kombinasi.
 3. Menghitung nilai permutasi.
 4. Menghitung nilai perpangkatan X^n
 5. Menghitung nilai deret angka $1+2+3+4+5+6+....$
 6. Menghitung nilai deret angka $2+4+6+8+10+....$
 7. Menghitung nilai deret angka $1+3+5+7+9+....$
- dengan menggunakan metode rekursif.
- Penjelasan mengenai algoritma permasalahan tersebut terdapat pada slide-slide berikutnya

Tugas rekursif II (TI1 pagi)

Silahkan kerjakan soal latihan yang terdapat pada powerpoint [rekursif II](#) pada slide ke 28 dengan ketentuan sebagai berikut:

1. Dikerjakan berkelompok sesuai dengan kelompok yang sudah dibentuk.
2. Buat program dalam bentuk iteratif dan [rekursif](#).
3. Tugas dikumpulkan paling lambat hari jumat, 16 Maret 2018, jam 08.30.

Selamat mengerjakan.

Jawaban

1. Membuat Barisan Bilangan Fibonacci

Rumus Matematika Fibonacci :

Fibonacci Math Rekursif

$$Fibonacci(n) \begin{cases} 1 & n=1 \\ 1 & n=2 \\ Fibonacci(n-1)+Fibonacci(n-2) & n>1 \end{cases}$$

A. Source Code Rekursif Fibonacci :

```
// Algoritma Fibonacci
#include <iostream>

using namespace std;

// Fungsi Fibonacci
int Fibonacci (int n)
{
    // Proses
    if (n <= 2)
        return 1;
    else if (n > 1)
        return Fibonacci(n-1) + Fibonacci(n-2);
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Fibonacci
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 0 || n == 0)
```

```

        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    else
    {
        // Output Success
        cout << "\nFibonacci n(" << n << ") = " << Fibonacci(n) << endl;
    }
}

```

B. Source Code Iteratif Fibonacci :

```

// Algoritma Fibonacci
#include <iostream>

using namespace std;

// Fungsi Fibonacci
int Fibonacci (int n)
{
    // Proses
    int i, hasil = 1, proses_1 = 1, proses_2 = 1;

    for (i = 2; i < n; i++)
    {
        hasil = proses_1 + proses_2;
        proses_1 = proses_2;
        proses_2 = hasil;
    }
    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Fibonacci
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 0 || n == 0)
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    else
    {
        // Output Success

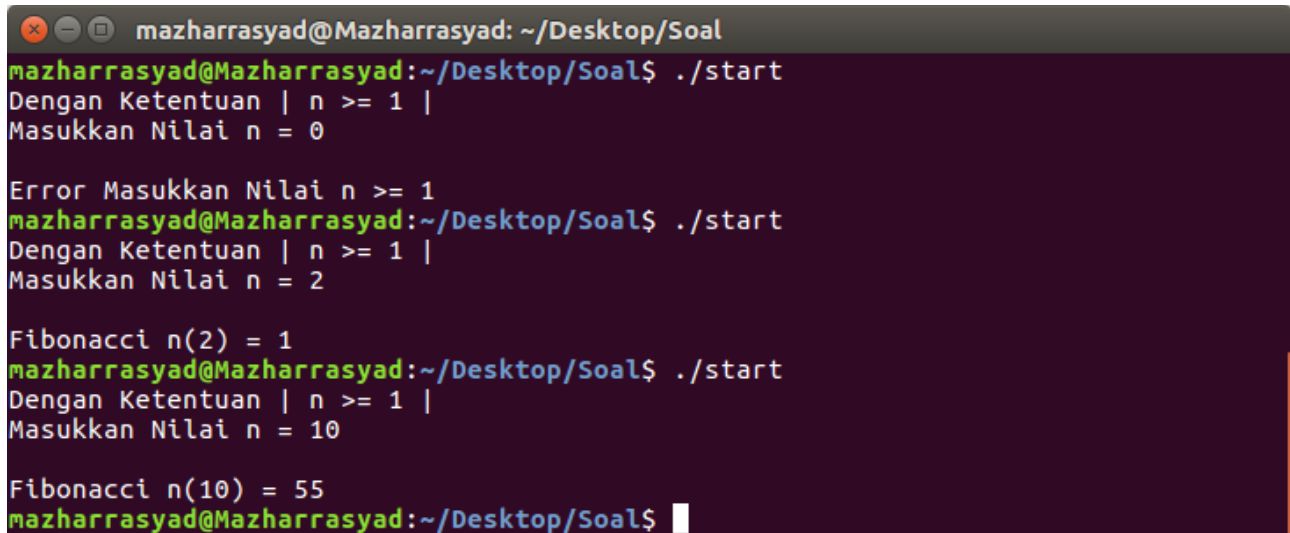
```

```

        cout << "\nFibonacci n(" << n << ") = " << Fibonacci(n) << endl;
    }
}

```

Contoh Hasil Rekursif dan Iteratif Fibonacci :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 0

Error Masukkan Nilai n >= 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 2

Fibonacci n(2) = 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 10

Fibonacci n(10) = 55
mazharrasyad@Mazharrasyad:~/Desktop/Soal$

```

Rekursif Fibonacci Dengan Proses :

```

// Algoritma Fibonacci
#include <iostream>

using namespace std;

// Fungsi Fibonacci
int Fibonacci (int n)
{
    // Proses
    if (n <= 2) // Proses Rumus Fibonacci
        return 1;
    else if (n > 1) // Proses Fibonacci
        return Fibonacci(n-1) + Fibonacci(n-2); // Proses Pertambahan Dari 1 Nilai Sebelumnya Dan 2 Nilai Sebelumnya

    /*
    Contoh Penjelasan Algoritma :

    Dengan Ketentuan n > 1
    Jika Nilai n = 4

    Maka Nilai n Dimasukkan Kedalam Fungsi Fibonacci(n)

    Fibonacci(n) = Fibonacci(n-1) + Fibonacci(n-2)

    Berikut Penjelasannya :
    Fibonacci(4) = Fibonacci(4-1) + Fibonacci(4-2)
    Fibonacci(4) = Fibonacci(3) + Fibonacci(2)

```

Dan Kemudian Mencari Nilai Fibonacci(3) + Fibonacci(2) Dengan Menggunakan Fungsi Yang Sama Secara Berulang

Fibonacci(3) = Fibonacci(3-1) + Fibonacci (3-2)

Fibonacci(3) = Fibonacci(2) + Fibonacci(1)

Fibonacci(2) = Fibonacci(2-1) + Fibonacci (2-2)

Fibonacci(2) = Fibonacci(1) + Fibonacci (0)

Dan Kemudian Mencari Nilai Fibonacci(1) dan Fibonacci(0)

Fibonacci(1) = 1

Fibonacci(0) = 1

Khusus Untuk n = 1 dan n = 2 Bernilai 1 Karena Rumus Fibonacci Sedangkan Sisanya Dijumlahkan

Maka Hasilnya Kemudian Dijumlahkan

Fibonacci(2) = Fibonacci(1) + Fibonacci (0) -> 1 "Karena n = 2"

Fibonacci(3) = Fibonacci(2) + Fibonacci(1) -> 1 + 1 = 2

Fibonacci(4) = Fibonacci(3) + Fibonacci(2) -> 2 + 1 = 3

Maka Hasil Fibonacci Dari Nilai n = 4 Adalah 3

*/

}

// Program Utama

int main()

{

// Deklarasi

int n,i; // Variabel

cout << "----- Algoritma Fibonacci -----" << endl << endl;

// Input

cout << "Dengan Ketentuan | n >= 1 |" << endl; // Ketentuan Bilangan Fibonacci

cout << "Masukkan Nilai n = ";

cin >> n; // Input Nilai n Dari Keyboard

// Verifikasi

if (n < 0 || n == 0)

cout << "\nError Masukkan Nilai n >= 1" << endl;

else

{

// Output

cout << "\nBerikut Barisan Bilangan Fibonacci(n) : " << endl;

for (i = 0; i < n; i++) // Output Perulangan Barisan Bilangan Fibonacci

{

cout << "Nilai n = " << i+1 << ", Fibonacci = " << Fibonacci(i+1);

if (n-(n-i) == 0 || n-(n-(i)) == 1)

cout << " Dari Rumus" << endl;

else

cout << " Dari " << Fibonacci(n-(n-i)) << " + " << Fibonacci(n-(n-(i-1))) << endl;

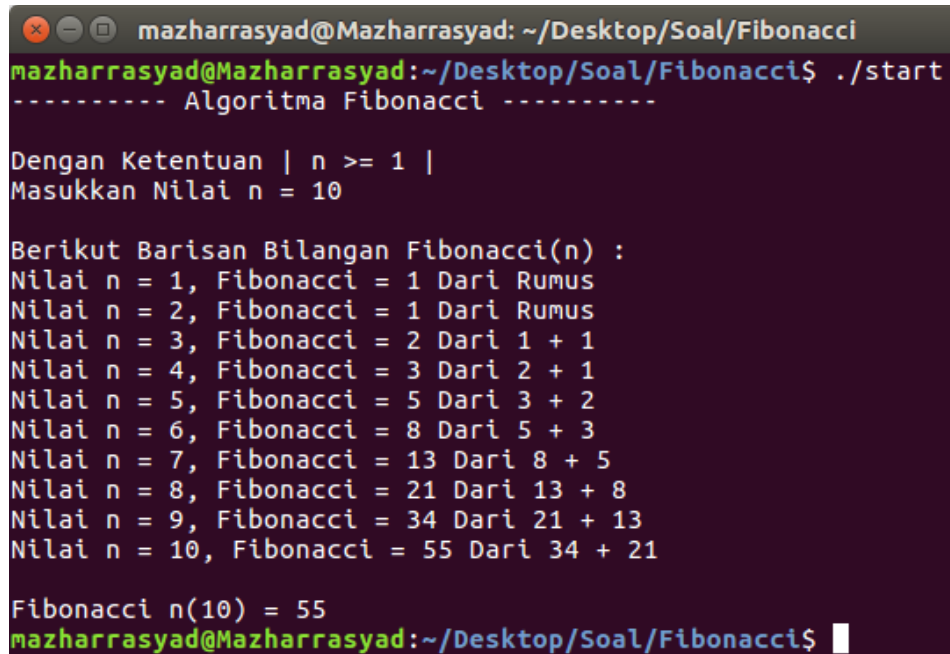
}

```

        cout << "\nFibonacci n(" << n << ") = " << Fibonacci(n) << endl; // Output Hasil Fibonacci Dari Nilai n
    }
}

```

Contoh Hasil Program :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Fibonacci
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Fibonacci$ ./start
----- Algoritma Fibonacci -----

Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 10

Berikut Barisan Bilangan Fibonacci(n) :
Nilai n = 1, Fibonacci = 1 Dari Rumus
Nilai n = 2, Fibonacci = 1 Dari Rumus
Nilai n = 3, Fibonacci = 2 Dari 1 + 1
Nilai n = 4, Fibonacci = 3 Dari 2 + 1
Nilai n = 5, Fibonacci = 5 Dari 3 + 2
Nilai n = 6, Fibonacci = 8 Dari 5 + 3
Nilai n = 7, Fibonacci = 13 Dari 8 + 5
Nilai n = 8, Fibonacci = 21 Dari 13 + 8
Nilai n = 9, Fibonacci = 34 Dari 21 + 13
Nilai n = 10, Fibonacci = 55 Dari 34 + 21

Fibonacci n(10) = 55
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Fibonacci$

```

2. Menghitung nilai kombinasi

Rumus Matematika Kombinasi :

Kombinasi Math Rekursif

$$Faktorial(n) \begin{cases} 1 & n \leq 1 \\ n * Faktorial(n-1) & n > 1 \end{cases}$$

$$Kombinasi(n) \begin{cases} 0 & n < r \\ Faktorial(n) / (Faktorial(r) * Faktorial(n-r)) & n \geq r \end{cases}$$

A. Source Code Rekursif Kombinasi :

```
// Algoritma Kombinasi
#include <iostream>

using namespace std;

// Fungsi Faktorial
int Faktorial (int n)
{
    // Proses
    if (n <= 1)
        return 1;
    else if (n > 1)
        return n * Faktorial(n-1);
}

// Fungsi Kombinasi
int Kombinasi (int n, int r)
{
    // Proses
    if (n < r)
        return 0;
    else if (n >= r)
        return Faktorial(n) / (Faktorial(n-r) * Faktorial(r));
}

// Program Utama
int main()
{
    // Deklarasi
    int n,r,i,j,k;

    // Ketentuan Bilangan Kombinasi
```



```

        cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl;

// Input
cout << "Masukkan Nilai n = ";
cin >> n;
cout << "Masukkan Nilai r = ";
cin >> r;

// Verifikasi
if (n < r || n <= 0 || r <= 0)
    // Output Error
    cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;
else
{
    // Output Success
    cout << "\nKombinasi C(" << n << ", " << r << ") = " << Kombinasi(n,r) << endl;
}
}

```

B. Source Code Iteratif Kombinasi :

```

// Algoritma Kombinasi
#include <iostream>

using namespace std;

// Fungsi Kombinasi
int Kombinasi (int n, int r)
{
    // Proses
    int i, hasil = 1, proses_1 = 1, proses_2 = 1;

    for (i = 0; i < n; i++)
    {
        hasil = hasil * (i + 1);
    }
    for (i = 0; i < (n-r); i++)
    {
        proses_1 = proses_1 * (i + 1);
    }
    for (i = 0; i < r; i++)
    {
        proses_2 = proses_2 * (i + 1);
    }
    hasil = hasil / (proses_1 * proses_2);
    return hasil;
}

// Program Utama
int main()
{

```

```

// Deklarasi
int n,r,i,j,k;

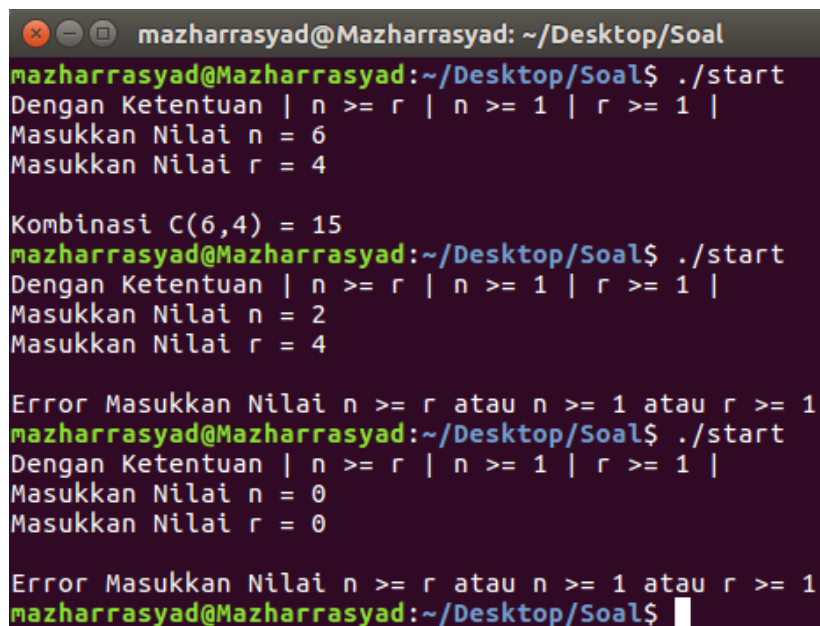
// Ketentuan Bilangan Kombinasi
cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl;

// Input
cout << "Masukkan Nilai n = ";
cin >> n;
cout << "Masukkan Nilai r = ";
cin >> r;

// Verifikasi
if (n < r || n <= 0 || r <= 0)
    // Output Error
    cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;
else
{
    // Output Success
    cout << "\nKombinasi C(" << n << ", " << r << ") = " << Kombinasi(n,r) << endl;
}
}

```

Contoh Hasil Rekursif dan Iteratif Kombinasi :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 6
Masukkan Nilai r = 4

Kombinasi C(6,4) = 15
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 2
Masukkan Nilai r = 4

Error Masukkan Nilai n >= r atau n >= 1 atau r >= 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 0
Masukkan Nilai r = 0

Error Masukkan Nilai n >= r atau n >= 1 atau r >= 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$

```

Rekursif Kombinasi Dengan Proses :

```

// Algoritma Kombinasi
#include <iostream>

using namespace std;

```

```

// Fungsi Faktorial
int Faktorial (int n)
{
    // Proses
    if (n <= 1) // Proses Rumus Faktorial
        return 1;
    else if (n > 1) // Proses Faktorial
        return n * Faktorial(n-1);
}

// Fungsi Kombinasi
int Kombinasi (int n, int r)
{
    // Proses
    if (n < r) // Proses Rumus Kombinasi
        return 0;
    else if (n >= r) // Proses Kombinasi
        return Faktorial(n) / (Faktorial(n-r) * Faktorial(r));
}

```

/*

Contoh Penjelasan Algoritma :

Dengan Ketentuan $| n > r | n >= 1 | r >= 1 |$

Jika Nilai $n = 3$ dan Nilai $r = 2$

Maka Nilai n Dimasukkan Kedalam Fungsi Kombinasi(n, r)

$Kombinasi(n, r) = Faktorial(n) / (Faktorial(n-r) * Faktorial(r))$

Karena Fungsi Kombinasi Memanggil Fungsi Faktorial Maka Proses Berlanjut Ke Fungsi Faktorial(n)

$Faktorial(n) = n * Faktorial(n-1)$

Berikut Penjelasannya :

$Kombinasi(3, 2) = Faktorial(3) / (Faktorial(3-2) * Faktorial(2))$

$Kombinasi(3, 2) = Faktorial(3) / (Faktorial(1) * Faktorial(2))$

Maka Fungsi Faktorial(n) Dijalankan Karena Bagian Dari Fungsi Kombinasi

$Faktorial(3) = 3 * Faktorial(3-1)$

$Faktorial(3) = 3 * Faktorial(2)$

$Faktorial(2) = 2 * Faktorial(2-1)$

$Faktorial(2) = 2 * Faktorial(1)$

$Faktorial(1) = 1 * Faktorial(1-1)$

$Faktorial(1) = 1 * Faktorial(0)$

Hasil Dari Faktorial(n)

$Faktorial(1) = 1$

$Faktorial(2) = 2 * Faktorial(1)$

$Faktorial(2) = 2 * 1$

$Faktorial(2) = 2$

$Faktorial(3) = 3 * Faktorial(2)$

Faktorial(3) = 3 * 2

Faktorial(3) = 6

Maka Dari Hasil Faktorial Tersebut Dapat Diberikan Nilai Kepada Fungsi Kombinasi

Kombinasi(3,2) = Faktorial(3) / (Faktorial(1) * Faktorial(2))

Kombinasi(3,2) = 6 / (1 * 2)

Kombinasi(3,2) = 6 / 2

Kombinasi(3,2) = 3

Maka Hasil Kombinasi Dari Nilai n = 3 Dan r = 2 Adalah 3

*/

// Program Utama

int main()

{

// Deklarasi

int n,r,i,j,k;

cout << "----- Algoritma Kombinasi -----" << endl << endl;

// Input

cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl; // Ketentuan Bilangan Kombinasi

cout << "Masukkan Nilai n = ";

cin >> n; // Input Nilai n Dari Keyboard

cout << "Masukkan Nilai r = ";

cin >> r; // Input Nilai r Dari Keyboard

// Verifikasi

if (n < r || n <= 0 || r <= 0)

cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;

else

{

// Output

cout << "\nBerikut Kombinasi C(n,r) : " << endl;

cout << "C(" << n << ", " << r << ") = " << n << "!" / ((" << n << " - " << r << ")! * " << r << "!) " << endl;

cout << "C(" << n << ", " << r << ") = " << n << "!" / (" << n-r << "!" * " << r << "!) " << endl;

cout << "C(" << n << ", " << r << ") = (";

for (i = 0; i < n; i++) // Output Perulangan Proses Kombinasi

{

cout << n-i;

if (n-i == 1)

{

cout << ") / (";

for (j = 0; j <= (n-r)-j; j++)

{

cout << (n-r)-j;

if ((n-r)-j == 1 || n-r == 0)

{

cout << ") * (";

for (k = 0; k < r; k++)

{

```

        cout << r-k;

        if (r-k == 1 || r-k == 0)
            cout << "));";
        else
            cout << " * ";
    }
}
else
    cout << " * ";
}
}
else
    cout << " * ";
}
cout << endl;
cout << "C(" << n << ", " << r << ") = " << Faktorial(n) << " / (" << Faktorial(n-r) << " * " << Faktorial(r) << ")" << endl;
cout << "C(" << n << ", " << r << ") = " << Faktorial(n) << " / (" << Faktorial(n-r) << " * " << Faktorial(r) << ")" << endl;
cout << "C(" << n << ", " << r << ") = " << Faktorial(n) << " / " << Faktorial(n-r) * Faktorial(r) << endl;
cout << "C(" << n << ", " << r << ") = " << Kombinasi(n,r);
cout << endl;

cout << "\nKombinasi C(" << n << ", " << r << ") = " << Kombinasi(n,r) << endl; // Output Hasil Kombinasi Dari Nilai n
    dan Nilai r
}
}

```

Contoh Hasil Program :

```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Kombinasi
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Kombinasi$ ./start
----- Algoritma Kombinasi -----

Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 6
Masukkan Nilai r = 4

Berikut Kombinasi C(n,r) :
C(6,4) = 6! / ((6 - 4)! * 4!)
C(6,4) = 6! / (2! * 4!)
C(6,4) = (6 * 5 * 4 * 3 * 2 * 1) / ((2 * 1) * (4 * 3 * 2 * 1))
C(6,4) = 720 / (2 * 24)
C(6,4) = 720 / (2 * 24)
C(6,4) = 720 / 48
C(6,4) = 15

Kombinasi C(6,4) = 15
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Kombinasi$

```

3. Menghitung nilai permutasi

Rumus Matematika Permutasi :

Permutasi Math Rekursif

$$Faktorial(n) \begin{cases} 1 & n \leq 1 \\ n * Faktorial(n-1) & n > 1 \end{cases}$$

$$Permutasi(n) \begin{cases} 0 & n < r \\ Faktorial(n) / Faktorial(n-r) & n \geq r \end{cases}$$

A. Source Code Rekursif Permutasi :

```
// Algoritma Permutasi
#include <iostream>

using namespace std;

// Fungsi Faktorial
int Faktorial (int n)
{
    // Proses
    if (n <= 1)
        return 1;
    else if (n > 1)
        return n * Faktorial(n-1);
}

// Fungsi Permutasi
int Permutasi (int n, int r)
{
    // Proses
    if (n < r)
        return 0;
    else if (n >= r)
        return Faktorial(n) / Faktorial(n-r);
}

// Program Utama
int main()
{
    // Deklarasi
```

```

int n,r;

// Ketentuan Bilangan Permutasi
cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl;

// Input
cout << "Masukkan Nilai n = ";
cin >> n;
cout << "Masukkan Nilai r = ";
cin >> r;

// Verifikasi
if (n < r || n <= 0 || r <= 0)
    // Output Error
    cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;
else
{
    // Output Success
    cout << "\nPermutasi P(" << n << ", " << r << ") = " << Permutasi(n,r) << endl;
}
}

```

B. Source Code Iteratif B :

```

// Algoritma Permutasi
#include <iostream>

using namespace std;

// Fungsi Permutasi
int Permutasi (int n, int r)
{
    int i, hasil = 1, proses_1 = 1;

    for (i = 0; i < n; i++)
    {
        hasil = hasil * (i + 1);
    }
    for (i = 0; i < (n-r); i++)
    {
        proses_1 = proses_1 * (i + 1);
    }
    hasil = hasil / proses_1;
    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n,r;

```

```

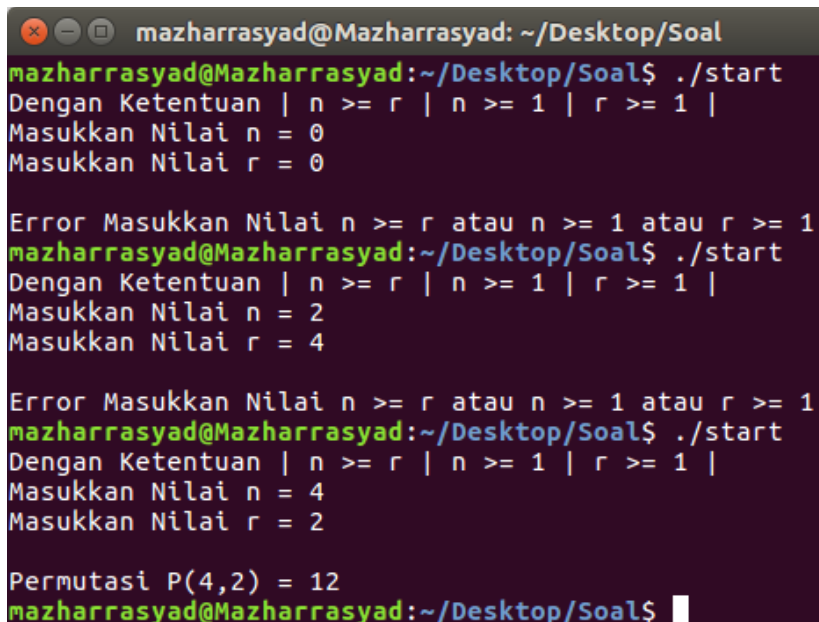
// Ketentuan Bilangan Permutasi
cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl;

// Input
cout << "Masukkan Nilai n = ";
cin >> n;
cout << "Masukkan Nilai r = ";
cin >> r;

// Verifikasi
if (n < r || n <= 0 || r <= 0)
    // Output Error
    cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;
else
{
    // Output Success
    cout << "\nPermutasi P(" << n << ", " << r << ") = " << Permutasi(n,r) << endl;
}
}

```

Contoh Hasil Rekursif dan Iteratif Permutasi :



```

mzharrasyad@Mzharrasyad: ~/Desktop/Soal
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 0
Masukkan Nilai r = 0

Error Masukkan Nilai n >= r atau n >= 1 atau r >= 1
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 2
Masukkan Nilai r = 4

Error Masukkan Nilai n >= r atau n >= 1 atau r >= 1
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 4
Masukkan Nilai r = 2

Permutasi P(4,2) = 12
mzharrasyad@Mzharrasyad:~/Desktop/Soal$

```

Rekursif Permutasi Dengan Proses :

```

// Algoritma Permutasi

```



```

#include <iostream>

using namespace std;

// Fungsi Faktorial
int Faktorial (int n)
{
    // Proses
    if (n <= 1) // Proses Rumus Faktorial
        return 1;
    else if (n > 1) // Proses Faktorial
        return n * Faktorial(n-1);
}

// Fungsi Permutasi
int Permutasi (int n, int r)
{
    // Proses
    if (n < r) // Proses Rumus Permutasi
        return 0;
    else if (n >= r) // Proses Permutasi
        return Faktorial(n) / Faktorial(n-r);
}

```

/*

Contoh Penjelasan Algoritma :

Dengan Ketentuan $| n > r | n >= 1 | r >= 1 |$

Jika Nilai $n = 3$ dan Nilai $r = 1$

Maka Nilai n Dimasukkan Kedalam Fungsi Permutasi(n, r)

$\text{Permutasi}(n, r) = \text{Faktorial}(n) / \text{Faktorial}(n-r)$

Karena Fungsi Permutasi Memanggil Fungsi Faktorial Maka Proses Berlanjut Ke Fungsi Faktorial(n)

$\text{Faktorial}(n) = n * \text{Faktorial}(n-1)$

Berikut Penjelasannya :

$\text{Permutasi}(3, 1) = \text{Faktorial}(3) / \text{Faktorial}(3-1)$

$\text{Permutasi}(3, 1) = \text{Faktorial}(3) / \text{Faktorial}(2)$

Maka Fungsi Faktorial(n) Dijalankan Karena Bagian Dari Fungsi Permutasi

$\text{Faktorial}(3) = 3 * \text{Faktorial}(3-1)$

$\text{Faktorial}(3) = 3 * \text{Faktorial}(2)$

$\text{Faktorial}(2) = 2 * \text{Faktorial}(2-1)$

$\text{Faktorial}(2) = 2 * \text{Faktorial}(1)$

$\text{Faktorial}(1) = 1 * \text{Faktorial}(1-1)$

$\text{Faktorial}(1) = 1 * \text{Faktorial}(0)$

Hasil Dari Faktorial(n)

$\text{Faktorial}(1) = 1$

```

Faktorial(2) = 2 * Faktorial(1)
Faktorial(2) = 2 * 1
Faktorial(2) = 2
Faktorial(3) = 3 * Faktorial(2)
Faktorial(3) = 3 * 2
Faktorial(3) = 6

```

Maka Dari Hasil Faktorial Tersebut Dapat Diberikan Nilai Kepada Fungsi Permutasi

```

Permutasi(3,1) = Faktorial(3) / Faktorial(2)
Permutasi(3,1) = 6 / 2
Permutasi(3,1) = 3

```

Maka Hasil Permutasi Dari Nilai n = 3 Dan r = 1 Adalah 3

```

*/

```

```

// Program Utama
int main()
{
    // Deklarasi
    int n,r,i,j,k;

    cout << "----- Algoritma Permutasi -----" << endl << endl;

    // Input
    cout << "Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |" << endl; // Ketentuan Bilangan Permutasi
    cout << "Masukkan Nilai n = ";
    cin >> n; // Input Nilai n Dari Keyboard
    cout << "Masukkan Nilai r = ";
    cin >> r; // Input Nilai r Dari Keyboard

    // Verifikasi
    if (n < r || n <= 0 || r <= 0)
        cout << "\nError Masukkan Nilai n >= r atau n >= 1 atau r >= 1" << endl;
    else
    {
        // Output
        cout << "\nBerikut Permutasi P(n,r) : " << endl;
        cout << "P(" << n << ", " << r << ") = " << n << "! / (" << n << " - " << r << ")!" << endl;
        cout << "P(" << n << ", " << r << ") = " << n << "! / " << n - r << "! " << endl;
        cout << "P(" << n << ", " << r << ") = (" << endl;
        for (i = 0; i < n; i++) // Output Perulangan Proses Permutasi
        {
            cout << n-i;

            if (n-i == 1 || n-i == 0)
            {
                cout << ") / (" << endl;
                if (n-r == 0)
                    cout << "1)";
                else
                {
                    for (j = 0; j < (n-r); j++)

```

```

        {
            cout << (n-r)-j;
            if ((n-r)-j == 1 || (n-r)-j == 0)
                cout << ")";
            else
                cout << " * ";
        }
    }
    else
        cout << " * ";
}
cout << endl;
cout << "P(" << n << ", " << r << ") = " << Faktorial(n) << " / " << Faktorial(n-r) << endl;
cout << "P(" << n << ", " << r << ") = " << Permutasi(n,r);
cout << endl;

cout << "\nPermutasi P(" << n << ", " << r << ") = " << Permutasi(n,r) << endl; // Output Hasil Permutasi Dari Nilai n
dan Nilai r
    }
}

```

Contoh Hasil Program :

```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Permutasi
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Permutasi$ ./start
----- Algoritma Permutasi -----

Dengan Ketentuan | n >= r | n >= 1 | r >= 1 |
Masukkan Nilai n = 4
Masukkan Nilai r = 2

Berikut Permutasi P(n,r) :
P(4,2) = 4! / (4 - 2)!
P(4,2) = 4! / 2!
P(4,2) = (4 * 3 * 2 * 1) / (2 * 1)
P(4,2) = 24 / 2
P(4,2) = 12

Permutasi P(4,2) = 12
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Permutasi$

```

4. Menghitung nilai perpangkat X^n

Rumus Matematika Perpangkatan :

Perpangkatan Math Rekursif

$$\text{Perpangkatan}(x, n) \begin{cases} 1 & n=0 \\ x * \text{Perpangkatan}(x, n-1) & n>1 \end{cases}$$

A. Source Code Rekursif Perpangkatan :

```
// Algoritma Perpangkatan
#include <iostream>

using namespace std;

// Fungsi Perpangkatan
int Perpangkatan (int x, int n)
{
    // Proses
    if (n == 0)
        return 1;
    else if (n > 1)
        return x * Perpangkatan(x,n-1);
}

// Program Utama
int main()
{
    // Deklarasi
    int n,x;

    // Ketentuan Bilangan Perpangkatan
    cout << "Dengan Ketentuan | x >= 0 | n >= 0 |" << endl;

    // Input
    cout << "Masukkan Nilai x = ";
    cin >> x;
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (x < 0 || n < 0)
        // Output Error
        cout << "\nError Masukkan Nilai x >= 0 atau n >= 0" << endl;
    else
```

```

    {
        // Output Success
        cout << "\nPerpangkatan(" << x << ", " << n << ") = " << Perpangkatan(x,n) << endl;
    }
}

```

B. Source Code Iteratif Perpangkatan :

```

// Algoritma Perpangkatan
#include <iostream>

using namespace std;

// Fungsi Perpangkatan
int Perpangkatan (int x, int n)
{
    // Proses
    int i, hasil = 1;

    for (i = 0; i < n; i++)
    {
        hasil = hasil * x;
    }

    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n,x;

    // Ketentuan Bilangan Perpangkatan
    cout << "Dengan Ketentuan | x >= 0 | n >= 0 |" << endl;

    // Input
    cout << "Masukkan Nilai x = ";
    cin >> x;
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (x < 0 || n < 0)
        // Output Error
        cout << "\nError Masukkan Nilai x >= 0 atau n >= 0" << endl;
    else
    {
        // Output Success
        cout << "\nPerpangkatan(" << x << ", " << n << ") = " << Perpangkatan(x,n) << endl;
    }
}

```

}

Contoh Hasil Rekursif dan Iteratif Perpangkatan :

```
mazharrasyad@Mazharrasyad: ~/Desktop/Soal
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | x >= 0 | n >= 0 |
Masukkan Nilai x = 0
Masukkan Nilai n = 0

Perpangkatan(0,0) = 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | x >= 0 | n >= 0 |
Masukkan Nilai x = 2
Masukkan Nilai n = 5

Perpangkatan(2,5) = 32
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | x >= 0 | n >= 0 |
Masukkan Nilai x = 0
Masukkan Nilai n = 1

Perpangkatan(0,1) = 0
mazharrasyad@Mazharrasyad:~/Desktop/Soal$
```

Rekursif Perpangkatan Dengan Proses :

```
// Algoritma Perpangkatan
```

```
#include <iostream>
```

```
using namespace std;
```

```
// Fungsi Perpangkatan
```

```
int Perpangkatan (int x, int n)
```

```
{
```

```
    // Proses
```

```
    if (n == 0) // Proses Rumus Perpangkatan
```

```
        return 1;
```

```
    else if (n > 1) // Proses Perpangkatan
```

```
        return x * Perpangkatan(x,n-1);
```

```
}
```

```
/*
```

```
Contoh Penjelasan Algoritma :
```

```
Dengan Ketentuan | x >= 0 | n >= 0 |
```

```
Jika Nilai x = 3 dan Nilai n = 2
```

```
Maka Nilai n Dimasukkan Kedalam Fungsi Perpangkatan(x,n)
```

```
Perpangkatan(x,n) = x * Perpangkatan(x,n-1)
```

```
Berikut Penjelasannya:
```

$\text{Perpangkatan}(3,2) = 3 * \text{Perpangkatan}(3,2-1)$

$\text{Perpangkatan}(3,2) = 3 * \text{Perpangkatan}(3,1)$

Jika Hasilnya Belum Diketahui Maka Memanggil Fungsi Yang Sama Secara Berulang

$\text{Perpangkatan}(3,1) = 3 * \text{Perpangkatan}(3,1-1)$

$\text{Perpangkatan}(3,1) = 3 * \text{Perpangkatan}(3,0)$

Karena $n = 0$ Maka Hasilnya Adalah 1 Sesuai Ketentuan Yang Berlaku

$\text{Perpangkatan}(3,0) = 1$

Jika Sudah Didapatkan Salah Satu Hasilnya Maka Hasil Yang Lain Akan Didapatkan

$\text{Perpangkatan}(3,0) = 1$

$\text{Perpangkatan}(3,1) = 3 * \text{Perpangkatan}(3,0)$

$\text{Perpangkatan}(3,1) = 3 * 1$

$\text{Perpangkatan}(3,1) = 3$

$\text{Perpangkatan}(3,2) = 3 * \text{Perpangkatan}(3,1)$

$\text{Perpangkatan}(3,2) = 3 * 3$

$\text{Perpangkatan}(3,2) = 9$

Maka Hasil Perpangkatan Bilangan 3 Dengan Pangkat 2 Adalah 27

*/

// Program Utama

int main()

{

 // Deklarasi

 int n,x,i;

 cout << "----- Algoritma Perpangkatan -----" << endl << endl;

 // Input

 cout << "Dengan Ketentuan | x >= 0 | n >= 0 |" << endl; // Ketentuan Bilangan Perpangkatan

 cout << "Masukkan Nilai x = ";

 cin >> x;

 cout << "Masukkan Nilai n = ";

 cin >> n;

 // Verifikasi

 if (x < 0 || n < 0)

 cout << "\nError Masukkan Nilai x >= 0 atau n >= 0" << endl;

 else

 {

 // Output

 cout << "\nBerikut Perpangkatan(x,n) : " << endl;

 if (n == 0)

 {

 cout << "Perpangkatan " << x << " Pangkat " << n << " = " << Perpangkatan(x,n) << " Dari Rumus" << endl;

 }

 else

```

{
    cout << "Proses Perpangkatan " << x << " Pangkat " << n << " = ";

    for (i = 0; i < n; i++) // Output Perulangan Proses Perpangkatan
    {
        cout << x << "(" << i + 1 << ")";

        if ((i + 1) == n)
        {
            cout << "";

            if (n == 1)
                cout << " Dari Bilangan Itu Sendiri";
        }
        else
            cout << " * ";
    }
    cout << endl << endl;

    for (i = 0; i < n; i++)
    {
        cout << "Nilai x = " << x << " Pangkat " << i + 1 << " = " << Perpangkatan(x,n-(i+1)) << endl;
    }

    cout << "\nPerpangkatan " << x << " Pangkat " << n << " = " << Perpangkatan(x,n) << endl;
}
cout << "\nPerpangkatan(" << x << ", " << n << ") = " << Perpangkatan(x,n) << endl; // Output Hasil Perpangkatan
Dari Nilai x dan Nilai n
}
}

```

Contoh Hasil Program :

```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Perpangkatan
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Perpangkatan$ ./start
----- Algoritma Perpangkatan -----

Dengan Ketentuan | x >= 0 | n >= 0 |
Masukkan Nilai x = 2
Masukkan Nilai n = 4

Berikut Perpangkatan(x,n) :
Proses Perpangkatan 2 Pangkat 4 = 2(1) * 2(2) * 2(3) * 2(4)

Nilai x = 2 Pangkat 1 = 2
Nilai x = 2 Pangkat 2 = 4
Nilai x = 2 Pangkat 3 = 8
Nilai x = 2 Pangkat 4 = 16

Perpangkatan 2 Pangkat 4 = 16

Perpangkatan(2,4) = 16
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Perpangkatan$

```


5. Menghitung nilai deret angka 1+2+3+4+5+6+...

Rumus Matematika Deret Angka 1+2+3+4+5+6+... :

Deret $S=1+2+3+4+5+...+n$ Math Rekursif

$$S(n) \begin{cases} 1 & n=1 \\ n+S(n-1) & n>1 \end{cases}$$

A. Source Code Rekursif Deret Angka 1+2+3+4+5+6+... :

```
// Algoritma Deret
#include <iostream>

using namespace std;

// Fungsi Deret
int Deret_N (int n)
{
    // Proses
    if (n == 1)
        return 1;
    else if (n > 1)
        return n + Deret_N(n-1);
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    else
    {
        // Output Success
```

```

        cout << "\nDeret(" << n << ") = " << Deret_N(n) << endl; // Output Hasil Deret Dari Nilai n
    }
}

```

B. Source Code Iteratif Deret Angka 1+2+3+4+5+6+... :

```

// Algoritma Deret
#include <iostream>

using namespace std;

// Fungsi Deret
int Deret_N (int n)
{
    int i, hasil = 0;

    for (i = 0; i < n; i++)
    {
        hasil = hasil + (i + 1);
    }

    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
    {
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    }
    else
    {
        // Output Success
        cout << "\nDeret(" << n << ") = " << Deret_N(n) << endl; // Output Hasil Deret Dari Nilai n
    }
}

```

Contoh Hasil Rekursif dan Iteratif Deret Angka 1+2+3+4+5+6+... :

```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 6

Deret(6) = 21
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 0

Error Masukkan Nilai n >= 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 1

Deret(1) = 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$

```

Rekursif Deret Angka 1+2+3+4+5+6+... Dengan Proses :

```

// Algoritma Deret
#include <iostream>

```

```

using namespace std;

```

```

// Fungsi Deret
int Deret_N (int n)
{
    // Proses
    if (n == 1) // Proses Rumus Deret
        return 1;
    else if (n > 1) // Proses Deret
        return n + Deret_N(n-1);
}

```

```

/*
Contoh Penjelasan Algoritma :

```

```

Dengan Ketentuan | n >= 1 |
Jika Nilai n = 3

```

Maka nilai n dimasukkan kedalam fungsi Deret_N(n)

Deret_N(n) = n + Deret_N(n-1)

Berikut Penjelasannya :

Deret_N(3) = 3 + Deret_N(3-1)

Deret_N(3) = 3 + Deret_N(2)

Karena Deret_N(2) tidak diketahui maka harus dicari terlebih dahulu dengan fungsi Deret_N(n)

Deret_N(2) = 2 + Deret_N(2-1)

Deret_N(2) = 2 + Deret_N(1)

Dari ketentuan yang berlaku pada rumus deret jika $n = 1$ maka hasilnya adalah 1

Deret_N(1) = 1

Deret_N(2) = 2 + Deret_N(1)

Deret_N(2) = 2 + 1

Deret_N(2) = 3

Jika Deret_N(2) sudah diketahui maka Deret_N(3) dapat diketahui

Deret_N(2) = 3

Deret_N(3) = 3 + Deret_N(2)

Deret_N(3) = 3 + 3

Deret_N(3) = 6

Maka Hasil Deret Nilai $n = 3$ Adalah 6

*/

// Program Utama

int main()

{

 // Deklarasi

 int n,i,j;

 cout << "----- Algoritma Deret -----" << endl << endl;

 // Input

 cout << "Dengan Ketentuan | $n \geq 1$ |" << endl; // Ketentuan Bilangan Deret

 cout << "Masukkan Nilai n = ";

 cin >> n;

 // Verifikasi

 if (n < 1)

 cout << "\nError Masukkan Nilai $n \geq 1$ " << endl;

 else

 {

 // Output

 cout << "\nBerikut Deret(n) : " << endl;

 cout << "Proses Deret(" << n << ") = ";

 for (i = 0; i < n; i++) // Output Perulangan Proses Deret

 {

 cout << i + 1;

 if ((i + 1) == n)

 {

 cout << "";

 }

 else

 cout << " + ";

 }

 cout << endl << endl;

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

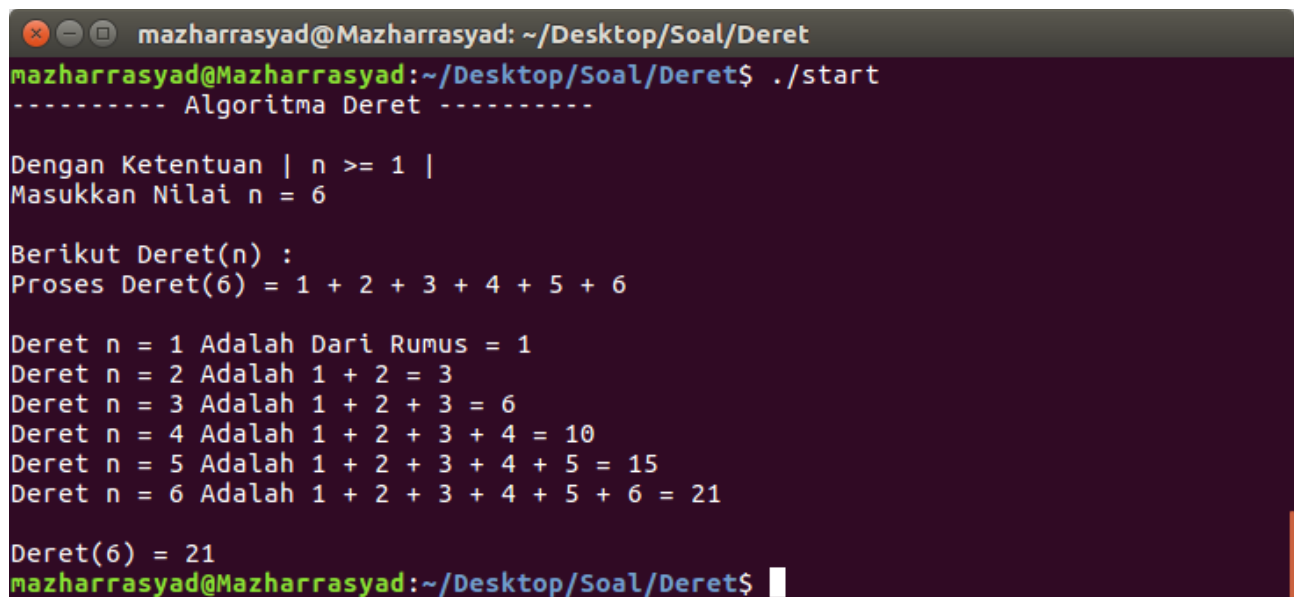
```

    {
        cout << "Deret n = " << i + 1 << " Adalah ";
        for (j = 0; j <= i; j++)
        {
            if (i + 1 == 1)
                cout << "Dari Rumus";
            else
            {
                cout << j + 1;
                if (j + 1 == i + 1)
                    cout << "";
                else
                    cout << " + ";
            }
        }
        cout << " = " << Deret_N(n-(i+1)) << endl;
    }

    cout << "\nDeret(" << n << ") = " << Deret_N(n) << endl; // Output Hasil Deret Dari Nilai n
}

```

Contoh Hasil Program :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Deret
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret$ ./start
----- Algoritma Deret -----

Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 6

Berikut Deret(n) :
Proses Deret(6) = 1 + 2 + 3 + 4 + 5 + 6

Deret n = 1 Adalah Dari Rumus = 1
Deret n = 2 Adalah 1 + 2 = 3
Deret n = 3 Adalah 1 + 2 + 3 = 6
Deret n = 4 Adalah 1 + 2 + 3 + 4 = 10
Deret n = 5 Adalah 1 + 2 + 3 + 4 + 5 = 15
Deret n = 6 Adalah 1 + 2 + 3 + 4 + 5 + 6 = 21

Deret(6) = 21
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret$

```

6. Menghitung nilai deret angka 2+4+6+8+10+...

Rumus Matematika Deret Angka 2+4+6+8+10+... :

Deret $S=2+4+6+8+10+...+2n$ Math Rekursif

$$S(n) \begin{cases} 2 & n=1 \\ 2*n+S(n-1) & n>1 \end{cases}$$

A. Source Code Rekursif Deret Angka 2+4+6+8+10+... :

```
// Algoritma Deret Genap
#include <iostream>

using namespace std;

// Fungsi Deret Genap
int Deret_Genap (int n)
{
    // Proses
    if (n == 1)
        return 2;
    else if (n > 1)
        return 2 * n + Deret_Genap(n-1);
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
    {
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    }
    else
    {
        // Output Success
```

```

        cout << "\nDeret Genap(" << n << ") = " << Deret_Genap(n) << endl;
    }
}

```

B. Source Code Iteratif Deret Angka 2+4+6+8+10+... :

```

// Algoritma Deret Genap
#include <iostream>

using namespace std;

// Fungsi Deret Genap
int Deret_Genap (int n)
{
    int i, hasil = 2;

    for (i = 2; i <= n; i++)
    {
        hasil = hasil + (2 * i);
    }

    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n,i,j,k;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
    {
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    }
    else
    {
        // Output Success
        cout << "\nDeret Genap(" << n << ") = " << Deret_Genap(n) << endl;
    }
}

```

Contoh Hasil Rekursif dan Iteratif Deret Angka 2+4+6+8+10+... :

```

mzharrasyad@Mzharrasyad: ~/Desktop/Soal
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 4

Deret Genap(4) = 20
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 0

Error Masukkan Nilai n >= 1
mzharrasyad@Mzharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 1

Deret Genap(1) = 2
mzharrasyad@Mzharrasyad:~/Desktop/Soal$

```

Rekursif Deret Angka 2+4+6+8+10+... Dengan Proses :

// Algoritma Deret Genap

#include <iostream>

using namespace std;

// Fungsi Deret Genap

int Deret_Genap (int n)

```

{
    // Proses
    if (n == 1) // Proses Rumus Deret Genap
        return 2;
    else if (n > 1) // Proses Deret Genap
        return 2 * n + Deret_Genap(n-1);
}

```

/*

Contoh Penjelasan Algoritma :

Dengan Ketentuan | n >= 1 |

Jika Nilai n = 3

Maka nilai n dimasukkan kedalam fungsi Deret_Genap(n)

Deret_Genap(n) = 2 * n + Deret_Genap(n-1)

Berikut Penjelasannya :

Deret_Genap(3) = 2 * 3 + Deret_Genap(3-1)

Deret_Genap(3) = 2 * 3 + Deret_Genap(2)

Deret_Genap(3) = 6 + Deret_Genap(2)

Karena Deret_Genap(2) tidak diketahui maka harus dicari terlebih dahulu dengan fungsi Deret_Genap(n)

Deret_Genap(2) = 2 * 2 + Deret_Genap(2-1)

$\text{Deret_Genap}(2) = 4 + \text{Deret_Genap}(1)$

Dari ketentuan yang berlaku pada rumus deret jika $n = 1$ maka hasilnya adalah 2

$\text{Deret_Genap}(1) = 2$

$\text{Deret_Genap}(2) = 4 + \text{Deret_Genap}(1)$

$\text{Deret_Genap}(2) = 4 + 2$

$\text{Deret_Genap}(2) = 6$

Jika $\text{Deret_Genap}(2)$ sudah diketahui maka $\text{Deret_Genap}(3)$ dapat diketahui

$\text{Deret_Genap}(2) = 6$

$\text{Deret_Genap}(3) = 6 + \text{Deret_Genap}(2)$

$\text{Deret_Genap}(3) = 6 + 6$

$\text{Deret_Genap}(3) = 12$

Maka Hasil Deret Genap Nilai $n = 3$ Adalah 12

*/

// Program Utama

int main()

{

 // Deklarasi

 int n,i,j,k;

 cout << "----- Algoritma Deret Genap -----" << endl << endl;

 // Input

 cout << "Dengan Ketentuan | $n \geq 1$ |" << endl; // Ketentuan Bilangan Deret

 cout << "Masukkan Nilai $n =$ ";

 cin >> n;

 // Verifikasi

 if ($n < 1$)

 cout << "\nError Masukkan Nilai $n \geq 1$ " << endl;

 else

 {

 // Output

 cout << "\nBerikut Deret Genap(n) : " << endl;

 cout << "Proses Deret Genap(" << n << ") = ";

 for ($i = 0$; $i < n$; $i++$) // Output Perulangan Proses Deret

 {

$k = k + 2$;

 cout << k;

 if ($((i + 1) == n)$)

 {

 cout << "";

 }

 else

 cout << " + ";

```

    }
    k = 0;
    cout << endl << endl;

    for (i = 0; i < n; i++)
    {
        cout << "Deret Genap n = " << i + 1 << " Adalah ";
        for (j = 0; j <= i; j++)
        {
            if (i + 1 == 1)
                cout << "Dari Rumus";
            else
            {
                k = k + 2;
                cout << k;

                if (j + 1 == i + 1)
                    cout << "";
                else
                    cout << " + ";
            }
        }
        k = 0;
        cout << " = " << Deret_Genap(n-(i+1)) << endl;
    }

    cout << "\nDeret Genap(" << n << ") = " << Deret_Genap(n) << endl; // Output Hasil Deret Genap Dari Nilai n
}

```

Contoh Hasil Program :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Deret Genap
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret Genap$ ./start
----- Algoritma Deret Genap -----

Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 4

Berikut Deret Genap(n) :
Proses Deret Genap(4) = 2 + 4 + 6 + 8

Deret Genap n = 1 Adalah Dari Rumus = 2
Deret Genap n = 2 Adalah 2 + 4 = 6
Deret Genap n = 3 Adalah 2 + 4 + 6 = 12
Deret Genap n = 4 Adalah 2 + 4 + 6 + 8 = 20

Deret Genap(4) = 20
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret Genap$

```

7. Menghitung nilai deret angka 1+3+5+7+9+...

Rumus Matematika Deret Angka 1+3+5+7+9+... :

Deret $S=1+3+5+7+9+...+2n-1$ Math Rekursif

$$S(n) \begin{cases} 1 & n=1 \\ (2*n)-1+S(n-1) & n>1 \end{cases}$$

A. Source Code Rekursif Deret Angka 1+3+5+7+9+... :

```
// Algoritma Deret Ganjil
#include <iostream>

using namespace std;

// Fungsi Deret Ganjil
int Deret_Ganjil (int n)
{
    // Proses
    if (n == 1)
        return 1;
    else if (n > 1)
        return (2 * n) - 1 + Deret_Ganjil(n-1);
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    else
    {
        // Output Success
```

```

        cout << "\nDeret Ganjil(" << n << ") = " << Deret_Ganjil(n) << endl; // Output Hasil Deret Ganjil Dari Nilai n
    }
}

```

B. Source Code Iteratif Deret Angka 1+3+5+7+9+... :

```

// Algoritma Deret Ganjil
#include <iostream>

using namespace std;

// Fungsi Deret Ganjil
int Deret_Ganjil (int n)
{
    int i, hasil = 1;

    for (i = 0; i <= n; i++)
    {
        hasil = hasil + ((2 * i) - 1);
    }

    return hasil;
}

// Program Utama
int main()
{
    // Deklarasi
    int n;

    // Ketentuan Bilangan Deret
    cout << "Dengan Ketentuan | n >= 1 |" << endl;

    // Input
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
    {
        // Output Error
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    }
    else
    {
        // Output Success
        cout << "\nDeret Ganjil(" << n << ") = " << Deret_Ganjil(n) << endl; // Output Hasil Deret Ganjil Dari Nilai n
    }
}

```

Contoh Hasil Rekursif dan Iteratif Deret Angka 1+3+5+7+9+... :

```
mazharrasyad@Mazharrasyad: ~/Desktop/Soal
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 0

Error Masukkan Nilai n >= 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 1

Deret Ganjil(1) = 1
mazharrasyad@Mazharrasyad:~/Desktop/Soal$ ./start
Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 5

Deret Ganjil(5) = 25
mazharrasyad@Mazharrasyad:~/Desktop/Soal$
```

Rekursif Deret Angka 1+3+5+7+9+... Dengan Proses :

```
// Algoritma Deret Ganjil
```

```
#include <iostream>
```

```
using namespace std;
```

```
// Fungsi Deret Ganjil
```

```
int Deret_Ganjil (int n)
```

```
{
```

```
    // Proses
```

```
    if (n == 1) // Proses Rumus Deret Ganjil
```

```
        return 1;
```

```
    else if (n > 1) // Proses Deret Ganjil
```

```
        return (2 * n) - 1 + Deret_Ganjil(n-1);
```

```
}
```

```
/*
```

```
Contoh Penjelasan Algoritma :
```

```
Dengan Ketentuan | n >= 1 |
```

```
Jika Nilai n = 3
```

```
Maka nilai n dimasukkan kedalam fungsi Deret_Ganjil(n)
```

```
Deret_Ganjil(n) = (2 * n) - 1 + Deret_Ganjil(n-1)
```

```
Berikut Penjelasannya :
```

```
Deret_Ganjil(3) = (2 * 3) - 1 + Deret_Ganjil(3-1)
```

```
Deret_Ganjil(3) = (2 * 3) - 1 + Deret_Ganjil(2)
```

```
Deret_Ganjil(3) = (6) - 1 + Deret_Ganjil(2)
```

```
Deret_Ganjil(3) = 5 + Deret_Ganjil(2)
```

```
Karena Deret_Ganjil(2) tidak diketahui maka harus dicari terlebih dahulu dengan fungsi Deret_Ganjil(n)
```

```

Deret_Ganjil(2) = (2 * 2) - 1 + Deret_Ganjil(2-1)
Deret_Ganjil(2) = (2 * 2) - 1 + Deret_Ganjil(1)
Deret_Ganjil(2) = (4) - 1 + Deret_Ganjil(1)
Deret_Ganjil(2) = 3 + Deret_Ganjil(1)

```

Dari ketentuan yang berlaku pada rumus deret jika $n = 1$ maka hasilnya adalah 1

```

Deret_Ganjil(1) = 1
Deret_Ganjil(2) = 3 + Deret_Ganjil(1)
Deret_Ganjil(2) = 3 + 1
Deret_Ganjil(2) = 4

```

Jika Deret_Ganjil(2) sudah diketahui maka Deret_Ganjil(3) dapat diketahui

```

Deret_Ganjil(2) = 4
Deret_Ganjil(3) = 5 + Deret_Ganjil(2)
Deret_Ganjil(3) = 5 + 4
Deret_Ganjil(3) = 9

```

Maka Hasil Deret Ganjil Nilai $n = 3$ Adalah 9
*/

```

/// Program Utama
int main()
{
    // Deklarasi
    int n,i,j,k;

    cout << "----- Algoritma Deret Ganjil -----" << endl << endl;

    // Input
    cout << "Dengan Ketentuan | n >= 1 |" << endl; // Ketentuan Bilangan Deret
    cout << "Masukkan Nilai n = ";
    cin >> n;

    // Verifikasi
    if (n < 1)
        cout << "\nError Masukkan Nilai n >= 1" << endl;
    else
    {
        // Output
        cout << "\nBerikut Deret Ganjil(n) : " << endl;
        cout << "Proses Deret Ganjil(" << n << ") = ";

        k = -1;
        for (i = 0; i < n; i++) // Output Perulangan Proses Deret
        {
            k = k + 2;
            cout << k;

```

```

        if ((i + 1) == n)
        {
            cout << "";
        }
        else
            cout << " + ";
    }
    cout << endl << endl;


    for (i = 0; i < n; i++)
    {
        cout << "Deret Ganjil n = " << i + 1 << " Adalah ";
        for (j = 0; j <= i; j++)
        {
            if (i + 1 == 1)
                cout << "Dari Rumus";
            else
            {
                k = k + 2;
                cout << k;

                if (j + 1 == i + 1)
                    cout << "";
                else
                    cout << " + ";
            }
        }
        k = -1;
        cout << " = " << Deret_Ganjil(n-(i+1)) << endl;
    }

    cout << "\nDeret Ganjil(" << n << ") = " << Deret_Ganjil(n) << endl; // Output Hasil Deret Ganjil Dari Nilai n
}

```

Contoh Hasil Program :



```

mazharrasyad@Mazharrasyad: ~/Desktop/Soal/Deret Ganjil
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret Ganjil$ ./start
----- Algoritma Deret Ganjil -----

Dengan Ketentuan | n >= 1 |
Masukkan Nilai n = 5

Berikut Deret Ganjil(n) :
Proses Deret Ganjil(5) = 1 + 3 + 5 + 7 + 9

Deret Ganjil n = 1 Adalah Dari Rumus = 1
Deret Ganjil n = 2 Adalah 1 + 3 = 4
Deret Ganjil n = 3 Adalah 1 + 3 + 5 = 9
Deret Ganjil n = 4 Adalah 1 + 3 + 5 + 7 = 16
Deret Ganjil n = 5 Adalah 1 + 3 + 5 + 7 + 9 = 25

Deret Ganjil(5) = 25
mazharrasyad@Mazharrasyad:~/Desktop/Soal/Deret Ganjil$

```

Referensi : www.google.com