

FUNGSI REKURSIF

FUNGSI REKURSIF

1. **Fungsi rekursif** merupakan sebuah metode perulangan yang terjadi akibat pengeksekusian suatu **fungsi**, di mana **fungsi** tersebut akan memanggil dirinya sendiri.
2. Hal ini memungkinkan bahwa **fungsi** rekursif akan terus memanggil dirinya sendiri tanpa batas atau dihentikan saat memenuhi kondisi tertentu.
3. Dalam beberapa kasus, **fungsi rekursif** bisa lebih efisien, tapi penulisannya memang tidak mudah dan sering terjadi error / infinity loop.

FUNGSI REKURSIF

bentuk umum fungsi rekursif memiliki statemen kondisional :

if *kondisi khusus tak dipenuhi*

then *panggil diri-sendiri dengan parameter yang sesuai*

else *lakukan instruksi yang akan dieksekusi bila kondisi khusus dipenuhi*

PERBEDAAN ITERATIF DAN REKURSIF

1. **Iteratif** menggunakan FOR, WHILE, DO-WHILE dan dapat berjalan pada program yang terdiri dari prosedur (Tidak terdapat fungsi).
2. **Rekursif** hanya menggunakan IF dan merupakan fungsi

Penggunaan Fungsi Rekursif

1. Penyelesaian sulit dilaksanakan secara iteratif.
2. Efisiensi dengan cara rekursif sudah memadai.
3. Efisiensi bukan masalah dibandingkan dengan kejelasan logika program

Bentuk dan Sifat Fungsi Rekursif

1. Ada bagian base case dan ada bagian general case
2. Paling sedikit mempunyai general base
3. Selalu dalam bentuk fungsi-fungsi
4. Selalu menggunakan statement percabangan

Kelebihan &Kekurangan Fungsi Rekursif

Kelebihan:

- Lebih efisien dan cepat dibandingkan proses secara iteratif
- Bentuk rekursif mudah dipahami alurnya

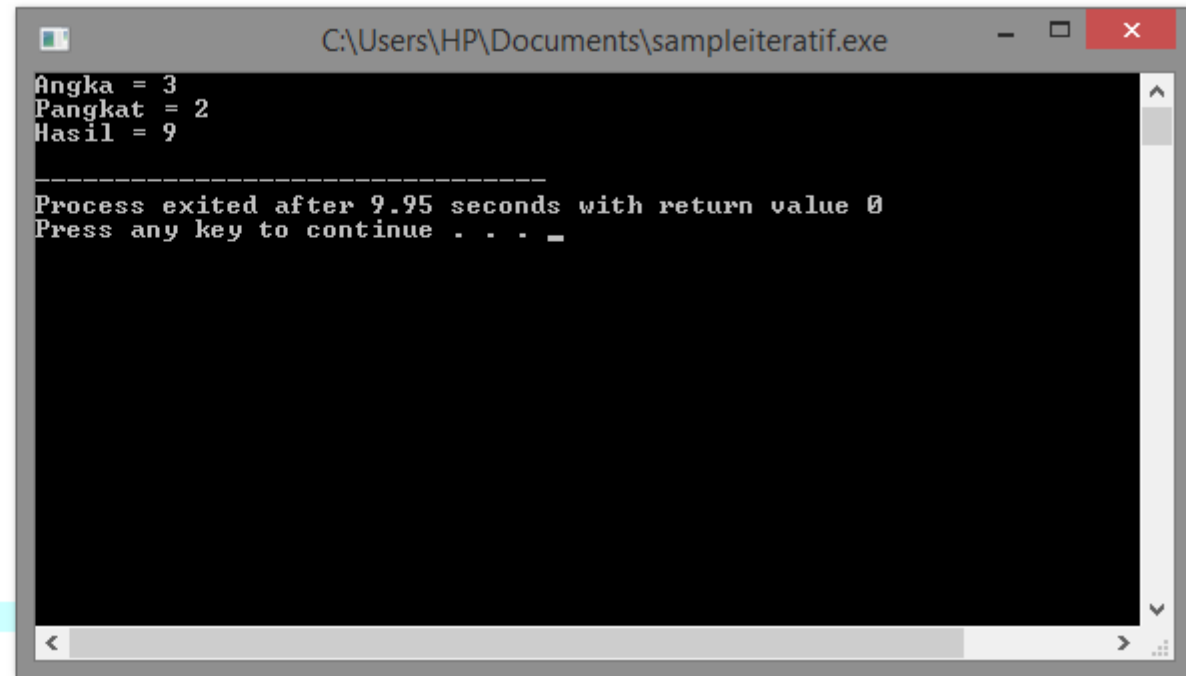
Kekurangan:

- Memakan memori lebih besar untuk menyimpan activation record dan variabel lokal. Activation record diperlukan waktu proses kembali kepada pemanggil.
- Memerlukan waktu yang lebih banyak untuk menangani activation record.

CONTOH FUNGSI NON REKURSIF

Non rekursif untuk pangkat

```
#include <stdio.h>
int fpangkat (int angka, int pangkat){
    int i;
    int hasil = 1;
    for (i = 1; i <= pangkat; i = i + 1){
        hasil = hasil * angka;
    }
    return hasil;
}
main(){
    int angka, pangkat;
    printf ("Angka = ");
    scanf ("%d", &angka);
    printf ("Pangkat = ");
    scanf ("%d", &pangkat);
    printf ("Hasil = %d\n", fpangkat (angka, pangkat));
}
```



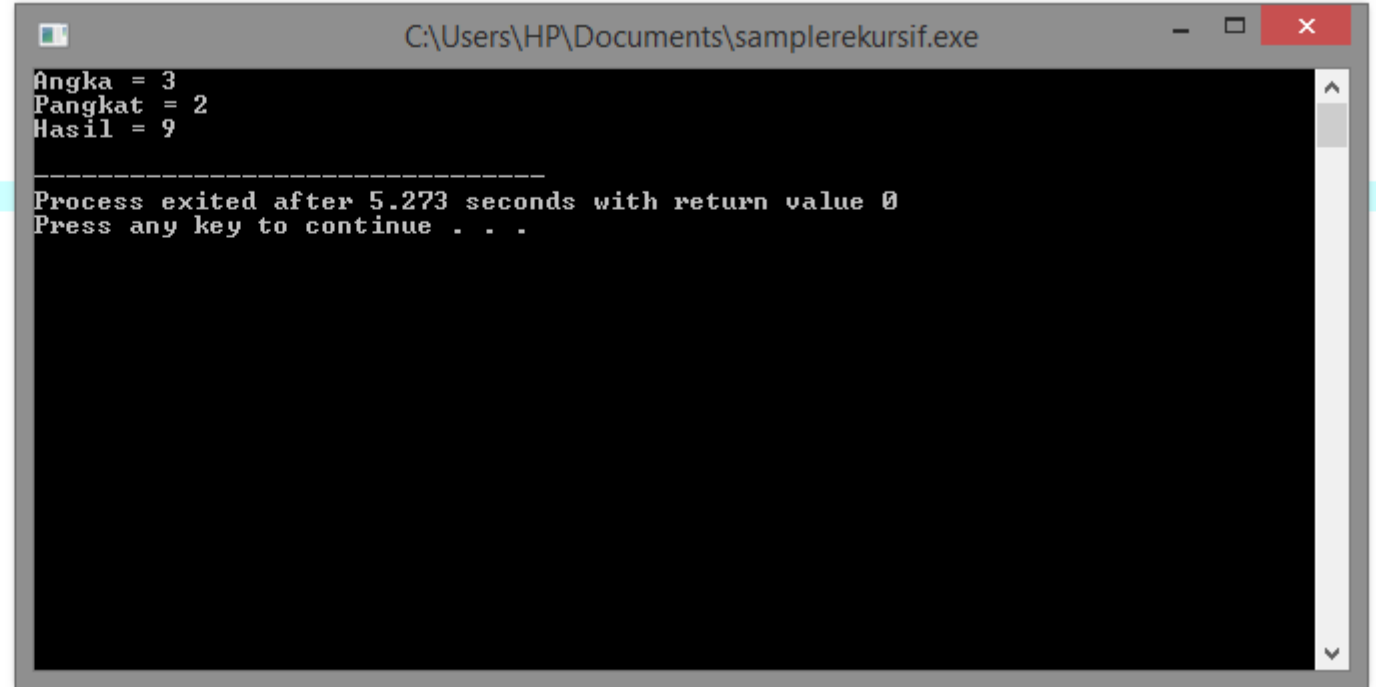
```
C:\Users\HP\Documents\sampleiteratif.exe
Angka = 3
Pangkat = 2
Hasil = 9

Process exited after 9.95 seconds with return value 0
Press any key to continue . . . _
```


CONTOH FUNGSI REKURSIF

Rekursif untuk pangkat

```
#include <stdio.h>
int fpangkat (int angka, int pangkat){
    if (pangkat == 0){
        return 1;
    }
    else {
        return fpangkat (angka, pangkat -1)*angka;
    }
}
main(){
    int angka, pangkat;
    printf ("Angka = ");
    scanf ("%d", &angka);
    printf ("Pangkat = ");
    scanf ("%d", &pangkat);
    printf ("Hasil = %d\n", fpangkat (angka, pangkat));
}
```



```
C:\Users\HP\Documents\samplerekursif.exe
Angka = 3
Pangkat = 2
Hasil = 9
-----
Process exited after 5.273 seconds with return value 0
Press any key to continue . . .
```

CONTOH FUNGSI NON REKURSIF

Deret fibonacci merupakan deretan angka yang disusun dari penjumlahan dua angka sebelumnya

Contoh Deret Fibonacci dengan non rekursif (iterative):

```
#include <iostream>

using namespace std;

int main()
{
    // rumus :  $F_n = F(n-1) + F(n-2)$ 
    // deret fibonacci : 0, 1, 1, 2, 3, 5, 8, dst.
    int n, Fn, Fn_min_1, Fn_min_2; // deklarasi variabel

    cout << " _____ " << endl << endl;
    cout << "    Program C++ Deret Fibonacci    " << endl;
    cout << " _____ " << endl << endl;

    cout << "    Berapa jumlah deret ? "; cin >> n;
    cout << endl << endl;
    cout << "    Fibonacci " << n << " deret : " << endl << endl;
    cout << " ";

    // devinisi value dari variabel
    Fn_min_1 = 1;
    Fn_min_2 = 0;

    // perulangan sebanyak jumlah deret
    for (int i = 1; i < n; i++){
        if (i == 1) cout << Fn_min_2 << " ";
        if (i == 2) cout << Fn_min_1 << " ";
        else {
            Fn = Fn_min_1 + Fn_min_2;
            Fn_min_2 = Fn_min_1;
            Fn_min_1 = Fn;
            cout << Fn << " ";
        }
    }

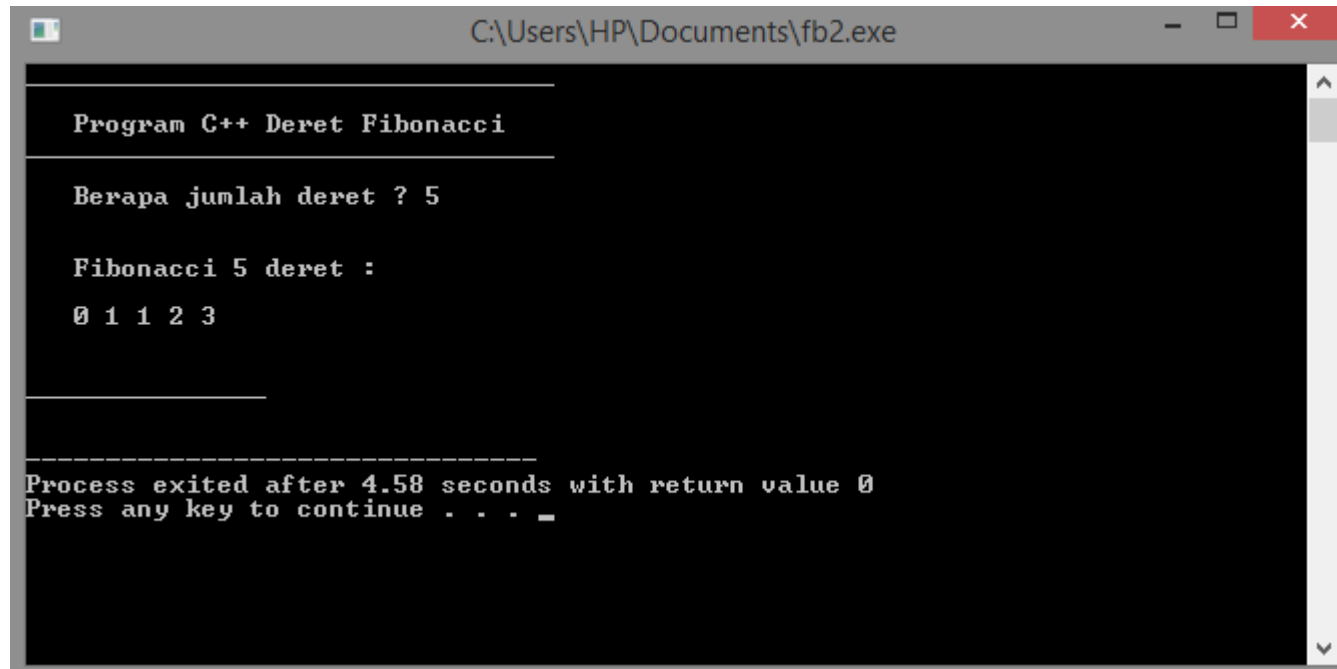
    cout << endl << endl << endl;
    cout << " _____ " << endl << endl;

    return 0;
}
```

CONTOH FUNGSI NON REKURSIF

Deret Fibonacci dengan Iteratif

Output



```
C:\Users\HP\Documents\fb2.exe

Program C++ Deret Fibonacci

Berapa jumlah deret ? 5

Fibonacci 5 deret :
0 1 1 2 3

-----
Process exited after 4.58 seconds with return value 0
Press any key to continue . . . _
```

CONTOH FUNGSI REKURSIF

Contoh Deret Fibonacci dengan fungsi rekursif:

```
#include <iostream>

using namespace std;

int F(int n){
    if (n == 0 || n == 1) return n;
    else {
        return F(n-1) + F(n-2); // rekursif
    }
}

int main()
{
    // rumus :  $F_n = F(n-1) + F(n-2)$ 
    // deret fibonacci : 0, 1, 1, 2, 3, 5, 8, dst.

    int n;

    cout << "_____ " << endl << endl;
    cout << "    Program C++ Deret Fibonacci " << endl;
    cout << "            Fungsi Rekursif " << endl;
    cout << "_____ " << endl << endl;

    cout << "    Berapa jumlah deret ? "; cin >> n;
    cout << endl << endl;
    cout << "    Fibonacci " << n << " deret : " << endl << endl;
    cout << "    ";

    // perulangan untuk memanggil fungsi F = fibonacci sebanyak n
    for (int i = 0; i < n; i++){
        cout << F(i) << " ";
    }

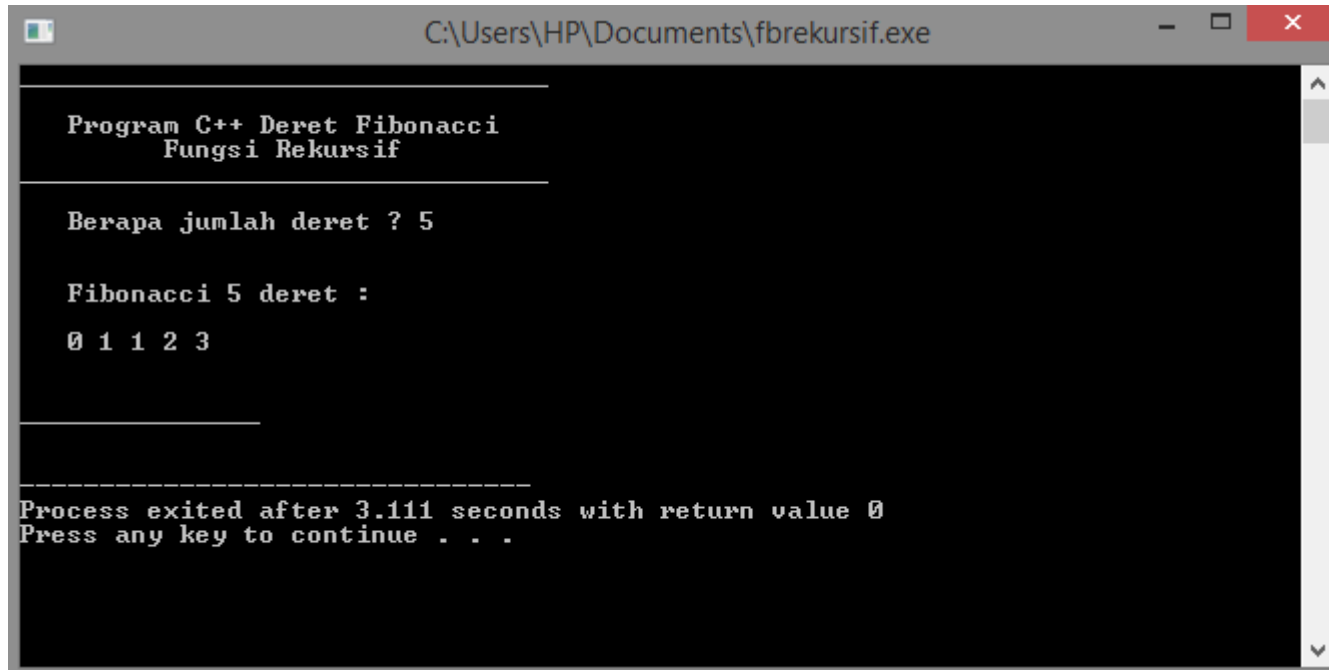
    cout << endl << endl << endl;
    cout << "_____ " << endl << endl;

    return 0;
}
```

CONTOH FUNGSI REKURSIF

Fibonacci

Output



```
C:\Users\HP\Documents\fbrekursif.exe

Program C++ Deret Fibonacci
Fungsi Rekursif

Berapa jumlah deret ? 5

Fibonacci 5 deret :
0 1 1 2 3

-----
Process exited after 3.111 seconds with return value 0
Press any key to continue . . .
```

CONTOH FUNGSI REKURSIF

Faktorial

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

long int faktorial (int A);

int main(){

    int r,hasil;

    cout<<"MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF"<<endl;
    cout<<endl;
    cout<<"Masukan Nilai = ";
    cin>>r;

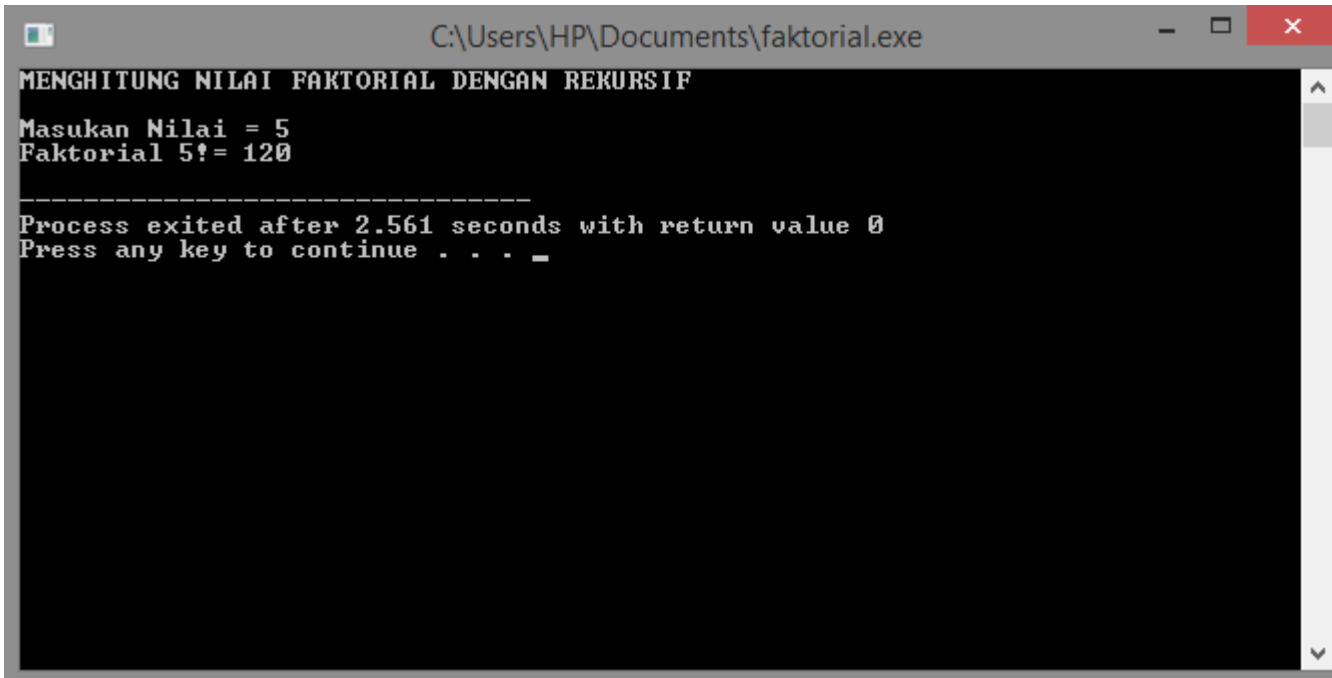
    hasil=faktorial(r);
    cout<<"Faktorial "<<r<<"!= "<<hasil<<endl;
}

long int faktorial (int A){
    if (A==1)
        return(A);
    else
        return (A*faktorial(A-1));
}
```

CONTOH FUNGSI REKURSIF

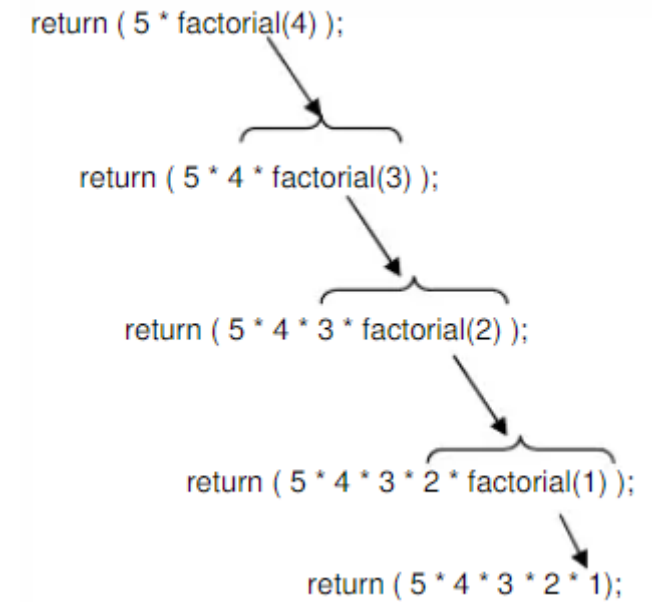
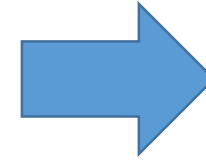
Faktorial

Output



```
C:\Users\HP\Documents\faktorial.exe
MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF
Masukan Nilai = 5
Faktorial 5!= 120

-----
Process exited after 2.561 seconds with return value 0
Press any key to continue . . . _
```



CONTOH FUNGSI REKURSIF

Permainan Menara Hanoi

```
#include <iostream>
#include <conio.h>
using namespace std;
void hanoi(int n, char dari, char bantu, char tujuan)
{
    if (n == 1)
        cout << "Pindahkan piring dari " << dari << " ke "
        << tujuan << "\n";
    else
    {
        hanoi(n-1, dari, tujuan, bantu);
        hanoi(1, dari, bantu, tujuan);
        hanoi(n-1, bantu, dari, tujuan);
    }
}

int main()
{
    int jum_piring;

    cout<<" Teknik Hanoi Pada c++" <<endl<<endl;
    cout << "Masukkan Jumlah piring: ";
    cin >> jum_piring;

    hanoi(jum_piring, 'A', 'B', 'C');
    getch();
}
```


CONTOH FUNGSI REKURSIF

Menara Hanoi



Output

```
C:\Users\HP\Documents\hanoi.exe
Teknik Hanoi Pada c++
Masukkan Jumlah piring: 3
Pindahkan piring dari A ke C
Pindahkan piring dari A ke B
Pindahkan piring dari C ke B
Pindahkan piring dari A ke C
Pindahkan piring dari B ke A
Pindahkan piring dari B ke C
Pindahkan piring dari A ke C
```

TUGAS

- Buat fungsi rekursif untuk
 - a.perkalian
 - b.pembagian

→Format Pengumpulan file dan folder tugas:

NPM_Nama_Kelas

→Link Gdrive cek di Ilmu

TERIMA KASIH