

## LAPORAN HASIL ANALISA PRAKTIKUM 6

### A. TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan rekursi?

Rekursif adalah suatu proses atau prosedur dari fungsi yang memanggil dirinya sendiri secara berulang-ulang. Karena proses dalam Rekursif ini terjadi secara berulang-ulang maka harus ada kondisi yang membatasi pengulangan tersebut, jika tidak maka proses tidak akan pernah berhenti sampai memori yang digunakan untuk menampung proses tersebut tidak dapat menampung lagi/penuh.

2. Tuliskan fungsi untuk menghitung nilai 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));
}
```

3. Tuliskan fungsi untuk menampilkan nilai fibonanci dari deret fibonanci

```
#include <iostream>
using namespace std;
int fibonacci(int m) {
    if (m == 0 || m ==1)
    {
        return m;
    }
    else
    {
        return (fibonacci(m-1) + fibonacci(m-2));
    }
}
int main() {
    int n, m= 0;
    cout << "Masukan Batas Deret Bilangan Fibonacci : ";
    cin >> n;
    cout << "Deret Fibonacci: ";
    for (int i = 1; i <= n; i++){
        cout << fibonacci(m) <<" ";
        m++;
    }
    return 0;
}
```

## B. PERCOBAAN

### Percobaan 1 : Fungsi rekursif untuk menghitung nilai 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));
}
```

#### Analisa

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

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. Using namespace std; merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
long int faktorial (int A);
```

*Long int faktorial (int A)* pendeklarasian fungsi faktorial dengan menggunakan parameter formal fungsi tipe data int variabel A. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi);

```
int main(){
```

*Int main()* { fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
int r,hasil;
```

*Int r,hasil;* pendeklarasian variabel r dan hasil tipe data int yang berada dalam fungsi main.

```
cout<<"MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF"<<endl;
cout<<endl;
```

*Cout<<"MENGHITUNG NILAI FAKTORIAL DENGAN REKRUSIF"<<endl;* menampilkan output dalam bentuk kalimat MENGHITUNG NILAI FAKTORIAL DENGAN REKRUSIF pada hasil program dan berpindah pada baris baru. *Cout<<endl;* akan menampilkan output berpindah ke baris baru pada hasil program.

```
cout<<"Masukan Nilai = ";
```

*Cout<<"Masukkan Nilai"<<endl;* menampilkan output dalam bentuk kalimat Masukkan Nilai pada hasil program.

```
cin>>r;
```

*Cin>>r;* proses memasukkan nilai variabel r menggunakan inputan keyboard.

```
hasil=faktorial(r);
```

*Hasil=faktorial(r);* pendefinisian variabel hasil yang merujuk pada fungsi factorial variabel r.

```
cout<<"Faktorial "<<r<<"!="<<hasil<<endl;
}
```

*Cout<<"Faktorial"<<r<<"!="<<hasil<<endl;* menampilkan output dalam bentuk kalimat Faktorial dengan menampilkan variabel r yang telah diinputkan oleh keyboard dan menampilkan tanda != beserta dengan variabel hasil pada hasil program. Kemudian berpindah pada baris baru. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi main.

```
long int faktorial (int A){
```

*Long int faktorial (int A){* fungsi long int factorial dengan menggunakan parameter formal fungsi tipe data int variabel A yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi long int faktorial akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi long int faktorial.

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

*If (A==1) return(A);* Kondisi jika nilai variabel A yang dimasukkan adalah 1 maka nilai balik (return value) adalah nilai itu sendiri. *Else return (A\*faktorial(A-1));* kondisi tidak (else) maka nilai balik yang dimasukkan akan dihitung menggunakan rumus factorial tersebut. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi long int faktorial.

```

MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF

Masukkan Nilai = 4
Faktorial 4!= 24

-----
Process exited after 4.23 seconds with return value 0
Press any key to continue . . .

```

Gambar 1. Output Hasil Program Menghitung Nilai Faktorial dengan Fungsi Rekrusif

## Percobaan 2 : Fungsi rekursi untuk menampilkan deret fibonanci

```

#include <iostream>
using namespace std;
int fibonacci(int m) {
    if (m == 0 || m ==1)
    {
        return m;
    }
    else
    {
        return (fibonacci(m-1) + fibonacci(m-2));
    }
}
int main() {
    int n, m= 0;
    cout << "Masukan Batas Deret Bilangan Fibonacci : ";
    cin >> n;
    cout << "Deret Fibonacci: ";
    for (int i = 1; i <= n; i++){
        cout << fibonacci(m) <<" ";
        m++;
    }
    return 0;
}

```

## Analisa

```

#include <iostream>
using namespace std;

```

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. Using namespace std; merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
int fibonacci(int m) {
```

*Int fibonacci (int m)* pendeklarasian fungsi fibonacci dengan menggunakan parameter formal fungsi tipe data int variabel m. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int fibonacci.

```
    if (m == 0 || m == 1)
    {
        return m;
    }
    else
    {
        return (fibonacci(m-1) + fibonacci(m-2));
    }
}
```

*If (m==0 || m==1){ return m;* Kondisi jika nilai variabel m yang dimasukkan adalah 0 atau 1 maka nilai balik (return value) adalah nilai itu sendiri. *}Else{ return (fibonacci(m-1) + fibonacci(m-2));* kondisi tidak (else) maka nilai balik yang dimasukkan akan dihitung menggunakan rumus fibonacci tersebut. Tanda *}* (kurung kurawal tutup) merupakan akhir dari fungsi int fibonacci.

```
int main(){
```

*Int main() {* fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
    int n, m= 0;
```

*Int n, m=0;* pendeklarasian variabel n dan m tipe data int yang berada dalam fungsi main. Variabel n digunakan untuk batas jumlah bilangan Fibonacci yang akan ditampilkan, dan variabel m digunakan untuk memasukkan nilai ke fungsi Fibonacci secara berurutan.

```
    cout << "Masukan Batas Deret Bilangan Fibonacci : ";
```

*Cout<<"Masukkan Batas Deret Bilangan Fibonacci :";* menampilkan output dalam bentuk kalimat Masukkan Batas Deret Bilangan Fibonacci : pada hasil program.

```
    cin >> n;
```

*Cin>>n;* proses memasukkan nilai variabel n menggunakan inputan keyboard.

```
cout << "Deret Fibonacci: ";
```

*Cout<<"Deret Fibonacci :"*; menampilkan output dalam bentuk kalimat Deret Fibonacci : pada hasil program.

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

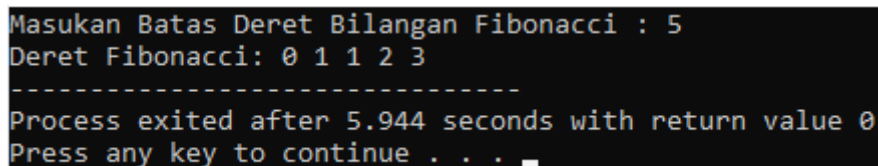
For (int i = 1; i<=n; i++){ perintah untuk memberi nilai awal perulangan yang dimulai dari i=1. Pendeklarasian variabel i tipe data int digunakan untuk perulangan. Perulangan akan berhenti jika nilai i <= inputan nilai n. *i++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for.

```
    cout << fibonacci(m) <<" ";  
    m++;  
}
```

*Cout<<Fibonacci(m) <<" "; m++;}* menampilkan output fungsi Fibonacci variabel m dan memberikan spasi. *M++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi for.

```
    return 0;  
}
```

return 0;} menyatakan hasil keluaran fungsi main adalah 0 atau untuk memberikan exit status yang berarti program berakhir dengan normal. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi main.



```
Masukan Batas Deret Bilangan Fibonacci : 5  
Deret Fibonacci: 0 1 1 2 3  
-----  
Process exited after 5.944 seconds with return value 0  
Press any key to continue . . .
```

Gambar 2. Output Hasil Program Menampilkan Deret Fibonacci dengan Fungsi Rekrusif

### Percobaan 3 : Fungsi rekursi untuk menentukan bilangan prima atau bukan prima

```
#include <iostream>
using namespace std;
int ambil(int bil, int i){
    if (i == 1) {
        return 1;
    }
    else if (bil % i == 0) {
        return 1 + ambil(bil, --i);
    } else {
        return 0 + ambil(bil, --i);
    }
}
int cek(int bil){
    if (bil > 1) {
        return (ambil(bil, bil) == 2);
    } else
        return false;
}
int main(){
    int bil;
    cout<<"Masukan Bilangan : ";
    cin>>bil;
    if (cek(bil)){
        cout<<"Bilangan Prima"<<endl;
    }else {
        cout<<"Bukan Bilangan Prima"<<endl;
    }
    return 0;
}
```

#### Analisa

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

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. Using namespace std; merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
int ambil(int bil, int i){
```

*Int ambil (int bil, int i){* pendeklarasian fungsi ambil dengan menggunakan parameter formal fungsi tipe data int variabel bil dan i. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int ambil.

```

    if (i == 1) {
        return 1;
    }
    else if (bil % i == 0) {
        return 1 + ambil(bil, --i);
    } else {
        return 0 + ambil(bil, --i);
    }
}

```

*If (i==1){ return 1;* Kondisi jika nilai variabel i yang dimasukkan adalah 1 maka nilai balik (return value) adalah nilai 1. *}Else if (bil%i==0){ return 1+ambil(bil, --i);* kondisi jika tidak (else if) apabila nilai variabel bil modulo i = 0 maka nilai balik yang dimasukkan akan dihitung menggunakan rumus bilangan prima tersebut. *}Else { return 0+ambil(bil, --i);* kondisi jika tidak (else) maka nilai balik yang dimasukkan akan dihitung menggunakan rumus bilangan prima tersebut. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi int ambil.

```

int cek(int bil){

```

*Int cek(int bil){* pendeklarasian fungsi cek dengan menggunakan parameter formal fungsi tipe data int variabel bil. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int cek.

```

    if (bil > 1) {
        return (ambil(bil, bil) == 2);
    } else
        return false;
}

```

*If (bil>1){ return (ambil(bil, bil)== 2);* Kondisi jika nilai variabel bil yang dimasukkan adalah (lebih dari) >1 atau bernilai benar maka nilai balik (return value) adalah nilai 2. *}Else return false;* kondisi tidak (else) maka nilai balik yang dimasukkan akan bernilai salah. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi int cek.

```

int main(){

```

*Int main() {* fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```

    int bil;

```

*Int bil;* pendeklarasian variabel bil tipe data int yang berada dalam fungsi main.

```

    cout<<"Masukan Bilangan : ";

```

*Cout<<"Masukkan Bilangan :";* menampilkan output dalam bentuk kalimat Masukkan



Bilangan pada hasil program.

```
cin>>bil;
```

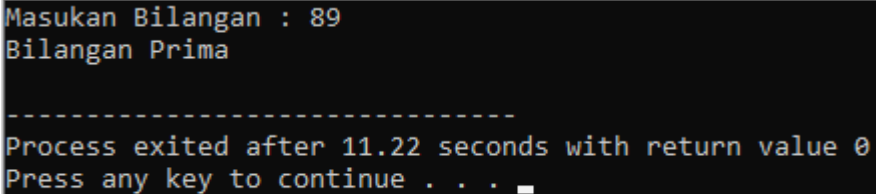
Cin>>bil; proses memasukkan nilai variabel bil menggunakan inputan keyboard.

```
if (cek(bil)){
    cout<<"Bilangan Prima"<<endl;
}else {
    cout<<"Bukan Bilangan Prima"<<endl;
}
```

If (cek(bil)){ cout<<"Bilangan Prima"<<endl; kondisi jika nilai yang dimasukkan bernilai benar di cek bil maka akan menampilkan output dalam bentuk kalimat Bilangan Prima pada hasil program. }else { cout<<"Bukan Bilangan Prima"<<endl;} kondisi tidak (else) nilai yang dimasukkan salah maka akan menampilkan menampilkan output dalam bentuk kalimat Bukan Bilangan Prima pada hasil program.

```
return 0;
}
```

return 0;} menyatakan hasil keluaran fungsi main adalah 0 atau untuk memberikan exit status yang berarti program berakhir dengan normal. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi main.



```
Masukan Bilangan : 89
Bilangan Prima
-----
Process exited after 11.22 seconds with return value 0
Press any key to continue . . .
```

Gambar 3. Output Hasil Program Menentukan Bilangan Prima dengan Fungsi Rekrusif

#### Percobaan 4 : Fungsi rekursi untuk menghitung pangkat

```
#include <iostream>
using namespace std;
long int pangkatrekursif(int x, int y);
int main(){
    int x,y;
    cout<<"FUNGSI REKURSIF UNTUK MENGHITUNG PANGKAT"<<endl;
    cout<<endl;
    cout<<"Masukan Nilai X = ";
    cin>>x; cout<<"Masukan Nilai Y = ";
    cin>>y;
    cout<<endl;
    cout<<x<<" Dipangkatkan "<<y<<" = "<<pangkatrekursif(x,y)<<endl;
}
long int pangkatrekursif(int x, int y){
    if (y==0)
        return 1 ;
    else
        return x * pangkatrekursif(x,y-1); }
```

## Analisa

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

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. *Using namespace std;* merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
long int pangkatrekursif(int x, int y);
```

*Long int pangkatrekursif (int x, int y)* pendeklarasian fungsi pangkat rekrusif dengan menggunakan parameter formal fungsi tipe data int variabel x dan y. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi);

```
int main(){
```

*Int main()* { fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
int x,y;
```

*Int x , y;* pendeklarasian variabel x dan y tipe data int yang berada dalam fungsi main.

```
cout<<"FUNGSI REKURSIF UNTUK MENGHITUNG PANGKAT"<<endl;
cout<<endl;
```

*Cout<<"FUNGSI REKRUSIF UNTUK MENGHITUNG PANGKAT"<<endl;* menampilkan output dalam bentuk kalimat *FUNGSI REKRUSIF UNTUK MENGHITUNG PANGKAT* pada hasil program dan berpindah pada baris baru. *Cout<<endl;* akan menampilkan output berpindah ke baris baru pada hasil program.

```
cout<<"Masukan Nilai X = ";
```

*Cout<<"Masukkan Nilai X = ";* menampilkan output dalam bentuk kalimat *Masukkan Nilai X =* pada hasil program.

```
cin>>x;
```

*Cin>>x;* proses memasukkan nilai variabel x menggunakan inputan keyboard.

```
cout<<"Masukan Nilai Y = ";
```

*Cout<<"Masukkan Nilai Y = "*; menampilkan output dalam bentuk kalimat Masukkan Nilai Y = pada hasil program.

```
cin>>y;  
cout<<endl;
```

*Cin>>y*; proses memasukkan nilai variabel y menggunakan inputan keyboard. *Cout<<endl*; akan menampilkan output berpindah ke baris baru pada hasil program.

```
cout<<x<<" Dipangkatkan "<<y<<" = "<<pangkatrekursif(x,y)<<endl;  
}
```

*Cout<<x<<"Dipangkatkan"<<y<<" = "<<pangkatrekursif(x,y)<<endl*; menampilkan output nilai variabel x dengan kalimat Dipangkatkan nilai variabel y yang sama-sama diinputkan oleh keyboard dan menampilkan tanda = beserta dengan nilai variabel pangkatrekursif(x,y) pada hasil program. Kemudian berpindah pada baris baru. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi main.

```
long int pangkatrekursif(int x, int y){
```

*Long int pangkatrekursif (int x, int y){* fungsi long int pangkat rekursif dengan menggunakan parameter formal fungsi tipe data int variabel x dan y yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi long int pangkat rekursif akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi long int pangkat rekursif.

```
if (y==0)  
    return 1 ;  
else  
    return x * pangkatrekursif(x,y-1); }
```

*If (y==1) return 1*; Kondisi jika nilai variabel y yang dimasukkan adalah 1 maka nilai balik (return value) adalah nilai 1. *Else return x \* pangkatrekursif (x,y-1)*; kondisi tidak (else) maka nilai balik yang dimasukkan akan dihitung menggunakan rumus pangkat tersebut. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi long int pangkatrekursif.

```
FUNGSI REKURSIF UNTUK MENGHITUNG PANGKAT  
  
Masukan Nilai X = 5  
Masukan Nilai Y = 3  
  
5 Dipangkatkan 3 = 125  
  
-----  
Process exited after 14.19 seconds with return value 0  
Press any key to continue . . .
```

Gambar 4. Output Hasil Program Menghitung Pangkat dengan Fungsi Rekursif

### C. LATIHAN

1. Buatlah program rekursif untuk menghitung segitiga Pascal !

```
F1          1
F2        1  1
F3      1  2  1
F4    1  3  3  1
F5  1  4  6  4  1
F6 1  5 10 10 5  1
```

```
#include <iostream>

using namespace std;

long faktorial(int n) {
    long z = 1;
    int i = 1;

    while(i<=n) {
        z=z*i;
        i++;
    }

    return z;
}

int main() {
    int a, i, j;

    cout<<"Masukkan nilai: ";
    cin >> a;

    for (i=0; i<a; i++) {
        for (j=0; j<a-i-1; j++){
            cout << " ";
        }

        for (j=0; j<=i; j++){
            cout << faktorial(i) / (faktorial(j) * faktorial(i - j)) << " ";
        }

        cout << endl;
    }

    return 0;
}
```

#### Analisa

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

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk

melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. *Using namespace std;* merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
long faktorial(int n) {
```

*Long faktorial (int n){* fungsi long faktorial dengan menggunakan parameter formal fungsi tipe data int variabel n yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi long faktorial akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi long faktorial.

```
long z = 1;  
int i = 1;
```

*Long z=1;* pendeklarasian nilai variabel z = 1 tipe data long yang berada dalam fungsi long faktorial. *Int i=1;* pendeklarasian nilai variabel i = 1 tipe data int yang berada dalam fungsi long faktorial.

```
while(i<=n) {
```

*While(i<=n){* Kondisi while akan berjalan ketika variabel counter i kurang dari samadengan variabel n. Ketika nilai variabel counter i sudah mencapai n, maka kondisi while (i <= n) tidak terpenuhi lagi (false), sehingga perulangan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi while.

```
z=z*i;  
i++;  
}
```

*Z=z\*i;* pendeklarasian rumus variabel z pada fungsi while. *I++;* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi while.

```
return z;  
}
```

*Return z;}* menyatakan hasil nilai balik (return value) fungsi long faktorial adalah nilai variabel z. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi long faktorial.

```
int main() {
```

*Int main()* { fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
Int a,i,j
```

*Int a, i, j;* pendeklarasian variabel a, i, j tipe data int yang berada dalam fungsi main. Variabel a digunakan untuk batas nilai yang dimasukkan untuk segitiga pascal, variabel i dan j digunakan untuk perulangan.

```
cout<<"Masukkan nilai: ";
```

*Cout<<"Masukkan nilai: ";* menampilkan output dalam bentuk kalimat Masukkan nilai: pada hasil program.

```
cin >> a;
```

*Cin>>a;* proses memasukkan nilai variabel a menggunakan inputan keyboard.

```
for (i=0; i<a; i++) {  
    for (j=0; j<a-i-1; j++){  
        cout << " ";  
    }  
}
```

*for (i=0; i<a; i++)* { perintah untuk memberi nilai awal perulangan yang dimulai dari i=0. Pendeklarasian variabel counter i tipe data int digunakan untuk perulangan pertama. Perulangan akan berhenti jika nilai i < inputan nilai a. *i++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for.

*for (j=0; j<a-i-1; j++)* { perintah untuk memberi nilai awal perulangan yang dimulai dari j=0. Pendeklarasian variabel counter j tipe data int digunakan untuk perulangan kedua. Perulangan akan berhenti jika nilai j < nilai a-i-1. *j++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti.

*cout << " ";* menampilkan spasi pada output hasil program. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi for.

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

*for (j=0; j<=i; j++)*{ perintah untuk memberi nilai awal perulangan yang dimulai dari j=0. Pendeklarasian variabel counter j tipe data int digunakan untuk perulangan. Perulangan akan berhenti jika nilai j <= nilai i. *j++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for.

```
cout << faktorial(i) / (faktorial(j) * faktorial(i - j)) << " ";
}
```

*cout << faktorial(i) / (faktorial(j) \* faktorial(i - j)) << " ";* menampilkan output hasil nilai variabel *faktorial(i) / (faktorial(j) \* faktorial(i - j))* dengan kalimat *Dipangkatkan nilai variabel y yang sama-sama diinputkan oleh keyboard dan menampilkan tanda = beserta dengan nilai variabel pangkatrekursif(x,y) pada hasil program.* Tanda *}* (kurung kurawal tutup) merupakan akhir dari fungsi *for*.

```
cout << endl;
}
```

*Cout<<endl;* akan menampilkan output berpindah ke baris baru pada hasil program. Tanda *}* (kurung kurawal tutup) merupakan akhir dari fungsi *for*.

```
return 0;
}
```

*return 0;* menyatakan hasil keluaran fungsi *main* adalah 0 atau untuk memberikan exit status yang berarti program berakhir dengan normal. Tanda *}* (kurung kurawal tutup) merupakan akhir dari fungsi *main*.

2. Buatlah program secara rekursif, masukkan jumlah N karakter dan cetak dalam semua kombinasi !

Jumlah karakter = 3

aaa aab aac aba abb abc aca acb acc baa bab bac bba bbb bbc  
bca bcb bcc caa cab cac cba cbb cbc cca ccb ccc BUILD

```
#include <bits/stdc++.h>
using namespace std;
void printLebarChar(char set[], string huruf,int n, int k)
{
    if (k == 0)
    {
        cout << (huruf) ;
        cout << " ";
        return;
    }
    for (int i = 0; i < n; i++)
    {
        string newHuruf;
        newHuruf = huruf + set[i];
        printLebarChar(set, newHuruf, n, k - 1);
    }
}

void inputLebarChar(char set[], int k,int n)
{
    printLebarChar(set, "", n, k);
}
int main()
{
    char set1[] = {'a', 'b', 'c'};
    int k = 3;
    cout << "Jumlah karakter : " ;
    cin >> k;

    inputLebarChar(set1, k, 3);
}
```

```
#include <bits/stdc++.h>
using namespace std;
```

*#include <bits/stdc++.h>* header yang akan meng-include semua standar library. File header *bits/stdc++.h* berfungsi agar kita tidak meng-includekan header library lainnya. *Using namespace std;* berkaitan dengan fungsi dari file header *iostream* karena berisi tentang perintah seperti *cout*, *cin*, dan *endl*. *Using namespace std;* merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
void printLebarChar(char set[], string huruf,int n, int k)
{
```

*Void printLebarChar(char set[], string huruf,int n, int k){* pendeklarasian fungsi void dengan menggunakan parameter fungsi tipe data char variabel *set[]*, tipe data string variabel *huruf*, tipe data int variabel *n* dan *k*. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi void.

```
if (k == 0)
{
```

*If (k == 0){* Kondisi jika nilai variabel *k* yang dimasukkan adalah 0. Tanda { (kurung kurawal buka) merupakan awal dari fungsi *if*.

```
    cout << (huruf) ;
    cout << " ";
    return;
}
```

*cout << (huruf) ; cout << " ";* menampilkan output nilai variabel *huruf* tipe data string pada hasil program dan memberikan spasi. *return;* mengembalikan hasil nilai balik (return value) fungsi *if*. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi *if*.

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

*for (int i = 0; i < n; i++){* perintah untuk memberi nilai awal perulangan yang dimulai dari *i=0*. Pendeklarasian variabel counter *i* tipe data int digunakan untuk perulangan. Perulangan akan berhenti jika nilai *i < nilai n*. *i++* perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi *for*.

```
    string newHuruf;
    newHuruf = huruf + set[i];
    printLebarChar(set, newHuruf, n, k - 1);
}
}
```

*String newHuruf;* pendeklarasian variabel *newhuruf* tipe data string yang berada dalam fungsi *for*. *newHuruf=huruf + set [i];* pendeklarasian rumus variabel new huruf pada



fungsi for. *printLebarChar(set, newHuruf, n, k - 1);*} pendeklarasian variabel lebar tipe data char yang berada dalam fungsi for. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi void.

```
void inputLebarChar(char set[], int k,int n)
{
```

*Void inputLebarChar(char set[], int k,int n){* pendeklarasian fungsi void dengan menggunakan parameter fungsi tipe data char variabel set[], tipe data int variabel n dan k. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi void.

```
    printLebarChar(set, "", n, k);
}
```

*printLebarChar(set, "", n, k);* pendeklarasian variabel lebar tipe data char yang berada dalam fungsi for.

```
int main() {
```

*Int main() {* fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan dihiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
char set1[] = {'a', 'b', 'c'};
int k = 3;
```

*char set1[] = {'a', 'b', 'c'};* pendeklarasian variabel set1[] tipe data char yang berada dalam fungsi main. *int k=3* pendeklarasian nilai variabel k tipe data int yang berada dalam fungsi main.

```
cout << "Jumlah karakter:" ;
```

*cout << "Jumlah karakter :"* ; menampilkan output dalam bentuk kalimat Jumlah karakter: pada hasil program.

```
cin >> k;
```

*cin >> k;* proses memasukkan nilai variabel k.

```
    inputLebarChar(set1, k, 3);
}
```

*inputLebarChar(setl, k, 3);* } pendeklarasian rumus input variabel lebar tipe data char.  
Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi void.

3. Buat program BinarySearch dengan Rekursif ! (data tentukan sendiri)

Data : 2,5,8,10,14,32, 35, 41, 67, 88, 90, 101, 109

Data yang dicari : 10

Data 10 berada pada indek ke - 3

```
#include <iostream>
#include <conio.h>
#include <iomanip>
using namespace std;

int data[7] = {1, 8, 2, 5, 4, 9, 7};
int cari;

void selection_sort()
{
    int temp, min, i, j;

    for(i=0; i<7;i++)
    {
        min = i;
        for(j = i+1; j<7; j++)
        {
            if(data[j]<data[min])
            {
                min=j;
            }
        }
        temp = data[i];
        data[i] = data[min];
        data[min] = temp;
    }
}

void binarysearch()
{
    //searching
    int awal, akhir, tengah, b_flag = 0;
    awal = 0;
    akhir = 7;
    while (b_flag == 0 && awal<=akhir)
    {
        tengah = (awal + akhir)/2;
        if(data[tengah] == cari)
        {
            b_flag = 1;
            break;
        }
        else if(data[tengah]<cari)
            awal = tengah + 1;
        else
            akhir = tengah -1;
    }

    if(b_flag == 1)
        cout<<"\nData ditemukan pada index ke-"<<tengah<<endl;
    else
        cout<<"\nData tidak ditemukan\n";
}
}
```

```

int main()
{
    cout<<"\t  'BINARY SEARCH'"<<endl;
    cout<<"\t===== "<<endl;
    cout<<"\nData      : ";
    //tampilkan data awal
    for(int x = 0; x<7; x++)
        cout<<setw(3)<<data[x];
    cout<<endl;

    cout<<"\nMasukkan data yang ingin Anda cari : ";
    cin>>cari;
    cout<<"\nData diurutkan : ";
    //urutkan data dengan selection sort
    selection_sort();
    //tampilkan data setelah diurutkan
    for(int x = 0; x<7;x++)
        cout<<setw(3)<<data[x];

    cout<<endl;

    binarysearch();

    _getche();
    return 0;
}

```

## Analisa

```

#include <iostream>
using namespace std;

```

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. Using namespace std; merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```

#include <conio.h>

```

*#include <conio.h>* memasukkan perintah untuk meng include kan file header conio.h sebelum proses compile berlangsung (preprocessing). File header conio.h berfungsi untuk menampilkan hasil antarmuka kepada pengguna dan melakukan operasi getch.

```

#include <iomanip>

```

*#include <iomanip>* memasukkan perintah untuk meng include kan file header iomanip sebelum proses compile berlangsung (preprocessing). File header iomanip berfungsi untuk melakukan operasi setw() .

```

int data[7] = {1, 8, 2, 5, 4, 9, 7};
int cari;

```

*int data[7] = {1, 8, 2, 5, 4, 9, 7};* pendeklarasian array (data) [7] tipe data int. *int cari;* pendeklarasian variabel cari tipe data int.

```
void selection_sort()
{
```

*void selection\_sort()*{ pendeklarasian fungsi void selection\_sort kombinasi antara sorting dan searching. Untuk setiap proses, akan dicari elemen-elemen yang belum diurutkan yang memiliki nilai terkecil atau terbesar akan dipertukarkan ke posisi yang tepat di dalam array. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi void.

```
int temp, min, i, j;
```

*int temp, min, i, j;* pendeklarasian variabel temp, min, I, j tipe data int yang berada dalam fungsi void.

```
for(i=0; i<7;i++)
{
```

*for(i=0; i<7;i++)*{ perintah untuk memberi nilai awal perulangan yang dimulai dari i=0. Pendeklarasian variabel counter i digunakan untuk perulangan pertama. Perulangan akan berhenti jika nilai i < 7. i++ perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for pertama.

```
min = i;
```

*min = i;* pendeklarasian nilai varabel min adalah nilai variabel I yang berada di dalam fungsi for pertama.

```
for(j = i+1; j<7; j++)
{
```

*for(j = i+1; j<7; j++)*{ perintah untuk memberi nilai awal perulangan yang dimulai dari j=i+1. Pendeklarasian variabel counter j digunakan untuk perulangan kedua. Perulangan akan berhenti jika nilai j < 7. j++ perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for kedua.

```
if(data[j]<data[min])
{
```

*if(data[j]<data[min])*{ kondisi jika akan berjalan ketika nilai variabel array data[j] < data[min]. Tanda { (kurung kurawal buka) merupakan awal dari fungsi if.

```
min=j;
}
}
```

*min=j;*}} pendeklarasian nilai varabel min adalah nilai variabel j yang berada di dalam fungsi if. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi if. Tanda { (kurung kurawal buka) merupakan awal dari fungsi for kedua.

```
Temp= data[i];  
Data [i]= data [min];  
Data [min]=temp;  
}  
}
```

*Temp= data[i];* pendeklarasian nilai varabel temp adalah nilai variabel data [i] yang berada di dalam fungsi for pertama. *Data [i]= data [min];* pendeklarasian nilai varabel data [i] adalah nilai variabel data [min] yang berada di dalam fungsi for pertama. *Data [min]=temp;* pendeklarasian nilai varabel data [min] adalah nilai variabel data [temp] yang berada di dalam fungsi for pertama. Tanda } (kurung kurawal tutup) merupakan awal dari fungsi for pertama. Tanda { (kurung kurawal buka) merupakan awal dari fungsi void.

```
void binarysearch()  
{  
    //searching
```

*void binarysearch()*{ pendeklarasian fungsi void binarysearch metode pencarian data harus diurutkan terlebih dahulu sebelum dilakukan proses pencarian. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi void. *//searching* merupakan sebuah komentar yang berada pada program dan tidak ditampilkan sebagai output.

```
int awal, akhir, tengah, b_flag = 0;  
awal = 0;  
akhir = 7;
```

*int awal, akhir, tengah, b\_flag = 0;* pendeklarasian variabel awal, akhir, tengah tipe data int dan nilai variabel b\_flag adalah 0 tipe data int yang berada dalam fungsi void. *awal = 0;* pendeklarasian nilai variabel awal adalah 0. *akhir = 7;* pendeklarasian nilai variabel akhir adalah 7.

```
while (b_flag == 0 && awal<=akhir)  
{
```

*while (b\_flag == 0 && awal<=akhir){* Kondisi while akan berjalan ketika variabel counter b\_flag = 0 dan awal kurang dari samadengan variabel akhir. Ketika nilai variabel counter tidak sesuai maka kondisi while tidak terpenuhi lagi (false), sehingga perulangan berhenti. Tanda { (kurung kurawal buka) merupakan awal dari fungsi while.

```
tengah = (awal + akhir)/2;
```

$tengah = (awal + akhir)/2$ ; pendeklarasian nilai varabel tengah adalah nilai variabel  $(awal+akhir) / 2$  yang berada di dalam fungsi while.

```
if(data[tengah] == cari)
{
```

*if(data[tengah] == cari)* kondisi jika akan berjalan ketika nilai variabel array data[tengah] = cari. Tanda { (kurung kurawal buka) merupakan awal dari fungsi if.

```
b_flag=1;
break;
}
```

*b\_flag=1*; pendeklarasian nilai varabel b-flag adalah 1 yang berada di dalam fungsi if.  
*Break*; pendeklarasian break berfungsi untuk menghentikan pernyataan pengulangan.

```
else if(data[tengah]<cari)
    awal = tengah + 1;
else
    akhir = tengah -1;
}
```

*else if(data[tengah]<cari) awal = tengah + 1*; kondisi jika tidak (else if) akan berjalan ketika nilai variabel array data[tengah] < cari, maka nilai varabel awal = tengah+1. Kondisi tidak (else) akan memasukkan nilai variabel akhir = tengah-1. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi if.

```
if(b_flag == 1)
    cout<<"\nData ditemukan pada index ke-"<<tengah<<endl;
else
    cout<<"\nData tidak ditemukan\n";
}
```

*if(b\_flag == 1) cout<<"\nData ditemukan pada index ke-"<<tengah<<endl*; Kondisi jika akan berjalan ketika nilai variabel b\_flag=1, maka akan menampilkan kalimat Data ditemukan pada index ke- serta nilai variabel tengah dan berpindah pada baris baru. *Else cout<<"\nData tidak ditemukan\n"*; kondisi tidak (else) akan menampilkan kalimat data tidak ditemukan. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi void.

```
int main() {
```

*Int main()* { fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan dihiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
cout<<"\t 'BINARY SEARCH'"<<endl;
```



*//urutkan data dengan selection sort* merupakan sebuah komentar yang berada pada program dan tidak ditampilkan sebagai output. *selection\_sort();* proses, akan dicari elemen-elemen yang belum diurutkan yang memiliki nilai terkecil atau terbesar akan dipertukarkan ke posisi yang tepat di dalam array.

```
//tampilkan data setelah diurutkan
for(int x = 0; x<7;x++)
    cout<<setw(3)<<data[x];

cout<<endl;
```

*//tampilkan data setelah diurutkan* merupakan sebuah komentar yang berada pada program dan tidak ditampilkan sebagai output. *for(int x = 0; x<7;x++)* perintah untuk memberi nilai awal perulangan yang dimulai dari x=0. Pendeklarasian variabel counter x tipe data int digunakan untuk perulangan. Perulangan akan berhenti jika nilai x< 7. x++ perulangan bersifat increment atau akan ditambah 1 setiap kali proses hingga sampai batas yang ditentukan maka proses akan berhenti. Jika perulangan bernilai benar maka akan menampilkan setw(3) yang berfungsi untuk mengatur lebar variabel dan nilai dari variabel array data[x]. *cout<<endl;* menampilkan output berpindah ke baris baru pada hasil program.

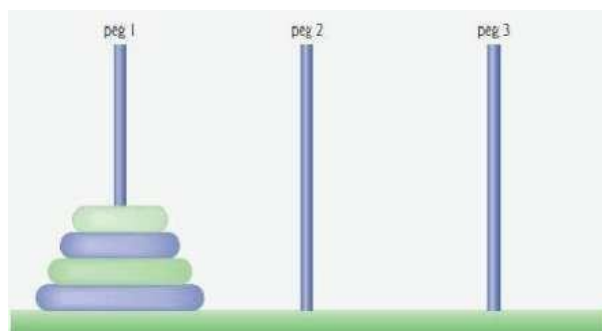
```
binarysearch();
_getche();
```

*binarysearch();* metode pencarian data harus diurutkan terlebih dahulu sebelum dilakukan proses pencarian. *\_getche();* memiliki fungsi hampir sama dengan getch(), tetapi inputan ditampilkan dalam window.

```
return 0;
}
```

return 0;} menyatakan hasil keluaran fungsi main adalah 0 atau untuk memberikan exit status yang berarti program berakhir dengan normal. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi main.

#### 4. Buatlah program rekursif untuk memecahkan permasalahan Menara Hanoi !



Gambar 1. Menara Hanoi



Program ini merupakan program untuk menampilkan pergerakan menara Hanoi yang merujuk pada class Menara Hanoi. Secara umum algoritma menara Hanoi adalah memindahkan sub menara Hanoi dengan n-1 pin dari n pin ke tiang perantara. Lalu memindahkan pin ke n ke tiang tujuan, lalu memindahkan sub menara Hanoi dengan n-1 pin yang ada di tiang perantara ke tiang tujuan. Stop casenya jika  $n = 1$ .

Jumlah disk : 3

Langkah-langkah nya adalah dengan :

1. Pindahkan disc 1 dari pasak A ke pasak C
2. Pindahkan disc 2 dari pasak A ke pasak B
3. Pindahkan disc 1 dari pasak C ke pasak B
4. Pindahkan disc 3 dari pasak A ke pasak C
5. Pindahkan disc 1 dari pasak B ke pasak A
6. Pindahkan disc 2 dari pasak B ke pasak C
7. Pindahkan disc 1 dari pasak A ke pasak C

```
#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();
}
```

## Analisa

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

*#include <iostream.h>* memasukkan perintah untuk meng include kan file header iostream sebelum proses compile berlangsung (preprocessing). File header iostream berfungsi untuk melakukan operasi cin dan cout. *Using namespace std;* berkaitan dengan fungsi dari file header iostream karena berisi tentang perintah seperti cout, cin, dan endl. *Using namespace std;* merupakan pernyataan (statement) yang selalu diakhiri dengan tanda ; (titik koma).

```
#include <conio.h>
```

*#include <conio.h>* memasukkan perintah untuk meng include kan file header conio.h sebelum proses compile berlangsung (preprocessing). File header conio.h berfungsi untuk menampilkan hasil antarmuka kepada pengguna dan melakukan operasi getch.

```
void hanoi(int n, char dari, char bantu, char tujuan)
{
```

*Void hanoi(int n, char dari, char bantu, char tujuan){* pendeklarasian fungsi void dengan menggunakan parameter fungsi tipe data char variabel dari, bantu, tujuan, dan tipe data int variabel n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi void.

```
    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);
    }
}
```

*if(n==1) cout<<"Pindahkan piring dari " << dari << " ke " << tujuan << "\n";* Kondisi jika akan berjalan ketika nilai variabel n=1, maka akan menampilkan kalimat Pindahkan piring dari, nilai variabel dari, menampilkan kalimat ke, nilai variabel tujuan, dan berpindah pada baris berikutnya. *else{hanoi(n-1, dari, tujuan, bantu);hanoi(1, dari, bantu, tujuan); hanoi(n-1, bantu, dari, tujuan);}* kondisi tidak (else) maka nilai variabel n yang dimasukkan akan dihitung menggunakan rumus hanoi tersebut. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi void.

```
int main() {
```

*Int main() {* fungsi main yang berisi statement-statement yang selalu diawali dengan tanda { (kurung kurawal buka) dan diakhiri dengan tanda } (kurung kurawal tutup). Fungsi main akan menjalankan statement-statement yang berada di dalamnya secara berurutan. Tanda { (kurung kurawal buka) merupakan awal dari fungsi main.

```
    int jum_piring;
```

*int jum\_piring;* pendeklarasian variabel jum\_piring tipe data int yang berada dalam fungsi main.

```
    cout<<"Teknik Hanoi Pada C++" <<endl<<endl;
    cout<<"Masukkan Jumlah Piring: ";
```

`cout<<"Teknik Hanoi Pada C++" <<endl<<endl;` menampilkan output dalam bentuk kalimat Teknik Hanoi Pada C++ pada hasil program. `cout<<"Masukkan Jumlah Piring :"`; menampilkan output dalam bentuk kalimat Masukkan Jumlah Piring : pada hasil program.

```
cin>>jum_piring;
```

`cin>>jum_piring;` proses memasukkan nilai variabel `jum_piring` menggunakan inputan keyboard.

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

`hanoi(jum_piring, 'A', 'B', 'C');` pendeklarasian variabel `hanoi` di dalam fungsi `main`.

```
getche ();  
}
```

`getche ();` memiliki fungsi hampir sama dengan `getch()`, tetapi inputan ditampilkan dalam window. Tanda `}` (kurung kurawal tutup) merupakan akhir dari fungsi `main`.

##### 5. Jelaskan proses rekursif untuk program dibawah ini !

```
void decToBin(int num)  
{  
    if (num > 0)  
    {  
        decToBin(num / 2);  
        cout << num % 2;  
    }  
}
```

##### Analisa

```
void decToBin(int num)  
{
```

`Void decTobin (int num){` pendeklarasian fungsi void dengan menggunakan parameter fungsi tipe `int` variabel `num`. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda `{` (kurung kurawal buka) merupakan awal dari fungsi `void decTobin`.

```
If (num > 0)  
{  
    decTobin (num/2);  
    cout << num % 2;  
}  
}
```

*If (num > 0){ decTobin (num/2);* Kondisi jika akan berjalan ketika nilai variabel num > 0, maka akan dioperasikan ke dalam decTobin. Tanda { (kurung kurawal buka) merupakan awal dari fungsi if. *Cout << num%2;}* menampilkan hasil output nilai variabel num modulu 2 pada hasil program. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi void. Tanda }(kurung kurawal tutup) merupakan aakhir dari fungsi if. Tanda }(kurung kurawal tutup) merupakan aakhir dari fungsi void decTobin.

6. Jelaskan proses rekursif untuk program dibawah ini !

```
boolean search(int[] x, int size, int n) {
    if (size > 0) {
        if (x[size-1] == n) {
            return true;
        } else {
            return search(x, size-1, n);
        }
    } return false;
}
```

### Analisa

```
boolean search(int[] x, int size, int n) {
```

*boolean search(int[] x, int size, int n)* { pendeklarasian fungsi Boolean merupakan jenis memori yang dapat mewakili satu nilai dari dua pilihan yaitu true dan false dengan menggunakan parameter fungsi tipe data int variabel array x, size, dan n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi boolean.

```
if (size > 0) {
```

*if (size > 0)* { kondisi jika akan berjalan ketika nilai variabel size >0. Tanda { (kurung kurawal buka) merupakan awal dari fungsi if.

```
if (x[size-1] == n) {
    return true;
```

*if (x[size-1] == n) { return true;* Kondisi jika nilai variabel x [size-1] adalah nilai n atau bernilai benar maka nilai balik (return value) adalah true.

```

    } else {
        return search(x, size-1, n);
    }
} return false;
}

```

*} else { return search(x, size-1, n); }* kondisi tidak (else) maka nilai balik (return value) akan dioperasikan ke dalam return search dan bernilai false. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi if. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi boolean.

7. Jelaskan proses rekursif untuk program dibawah ini !

```

boolean binarySearch(int[] x, int start, int end, int n) {
    if (end < start)
        return false;
    int mid = (start+end) / 2;
    if (x[mid] == n) {
        return true;
    } else {
        if (x[mid] < n) {
            return search(x, mid+1, end, n);
        } else {
            return search(x, start, mid-1, n);
        }
    }
}

```

### Analisa

```

boolean binarySearch(int[] x, int start, int end, int n) {

```

*boolean binarySearch(int[] x, int start, int end, int n) {* pendeklarasian fungsi Boolean binarysearch merupakan jenis memori yang dapat mewakili satu nilai dari dua pilihan yaitu true dan false dengan metode pencarian data harus diurutkan terlebih dahulu sebelum dilakukan proses pencarian. Menggunakan parameter fungsi tipe data int variabel array x, start, end, dan n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi Boolean binarysearch.

```

    if (end < start)
        return false;

```

*if (end < start) return false; if (x[size-1] == n) { return true;* Kondisi jika nilai end<start maka nilai balik (return value) adalah false.

```
int mid = (start+end) / 2;
```

*int mid = (start+end) / 2;* pendeklarasian tipe data int nilai varabel mid adalah (start+end)/2 yang berada di dalam fungsi Boolean binarysearch.

```
if (x[mid] == n) {  
    return true;
```

*if (x[mid] == n) { return true;* Kondisi jika nilai variabel x [mid] adalah nilai n atau bernilai benar maka nilai balik (return value) adalah true. Tanda { (kurung kurawal buka) merupakan awal dari fungsi if.

```
} else {  
    if (x[mid] < n) {  
        return search(x, mid+1, end, n);
```

*} else { if (x[mid] < n) { return search(x, mid+1, end, n);* kondisi jika tidak (else if) akan berjalan ketika x[mid] < n maka nilai balik (return value) akan dioperasikan ke dalam return search.

```
} else {  
    return search(x, start, mid-1, n);  
}  
}  
}
```

*} else { return search(x, start, mid-1, n);} } }* kondisi tidak (else) maka nilai balik (return value) akan dioperasikan ke dalam return search. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi if. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi boolean.

#### 8. Jelaskan proses rekursif untuk program dibawah ini dengan memanggil

mystery(2, 25) and mystery(3, 11)!

```
int mystery(int a, int b) {  
    if (b == 0)  
        return 0;  
    if (b % 2 == 0)
```

```

    return mystery(a+a, b/2);
return mystery(a+a, b/2) + a;
}

```

## Analisa

```
int mystery(int a, int b) {
```

*int mystery(int a, int b) {* pendeklarasian fungsi mystery dengan menggunakan parameter formal fungsi tipe data int variabel a dan b. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int mystery.

```

    if (b == 0)
        return 0;

```

*if (b == 0) return 0;* Kondisi jika nilai variabel b=0 bernilai benar maka nilai balik (return value) adalah 0.

```

    if (b % 2 == 0)
        return mystery(a+a, b/2);
    return mystery(a+a, b/2);+a
}

```

*if (b % 2 == 0) return mystery(a+a, b/2);* Kondisi jika nilai variabel b modulo 2 adalah 0 atau bernilai benar maka nilai balik (return value) akan dioperasikan ke dalam return mystery. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi int mystery.

## 9. Jelaskan proses rekursif untuk program dibawah ini dengan memanggil mystery(0, 8)!

```

int mystery(int a, int b) {
    if (a == b) cout<<a<<endl;
    else {
        int m1 = (a + b) / 2;
        int m2 = (a + b + 1) / 2;
        mystery(a, m1);
        mystery(m2, b);
    }
}

```

## Analisa

```
int mystery(int a, int b) {
```

*int mystery(int a, int b) {* pendeklarasian fungsi mystery dengan menggunakan parameter formal fungsi tipe data int variabel a dan b. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int mystery.

```
if (a == b) cout<<a<<endl;
```

*if (a == b) cout<<a<<endl;* Kondisi jika nilai variabel a sama dengan nilai variabel b atau bernilai benar maka akan menampilkan nilai variabel a dan berpindah pada baris baru.

```
else {  
    int m1 = (a+b)/2;  
    int m2 = (a+b+1)/2;
```

*else { int m1 = (a+b)/2 ;int m2 = (a+b+1)/2;* kondisi tidak (else) maka nilai akan dioperasikan ke dalam deklarasi variabel m1 dan m2 tipe data int.

```
    mystery(a, m1);  
    mystery(m2, b);  
}
```

*mystery(a, m1);mystery(m2, b);}}* pendeklarasian variabel mystery di dalam fungsi int mystery. Tanda } (kurung kurawal tutup) merupakan aakhir dari fungsi int mystery.

### 10. Jelaskan proses rekursif untuk program dibawah ini !

```
int f(int n) {  
    if (n == 0)  
        return 0;  
    if (n == 1)  
        return 1;  
    if (n == 2)  
        return 1;  
    return 2*f(n-2) + f(n-3);
```



## Analisa

```
int f(int n) {
```

*int f(int n) {* pendeklarasian fungsi f dengan menggunakan parameter formal fungsi tipe data int variabel n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int f.

```
    if (n == 0)
        return 0;
```

*if (n == 0) return 0;* Kondisi jika nilai variabel n=0 bernilai benar maka nilai balik (return value) adalah 0.

```
    if (n == 1)
        return 1;
```

*if (n== 1) return 1;* Kondisi jika nilai variabel n=1 bernilai benar maka nilai balik (return value) adalah 1.

```
    if (n == 2)
        return 1;
        return 2*f(n-2) + f(n-3);
```

*if (n== 2) return 1; return 2\*f(n-2) + f(n-3);* Kondisi jika nilai variabel n=2 bernilai benar maka nilai balik (return value) adalah 1 dan dioperasikan ke dalam return  $2*f(n-2) + f(n-3)$ .

11. Jelaskan proses rekursif untuk program dibawah ini dengan memanggil `square(5)`,  
`cube(5)`, `cube(123)`?

```
int square(int n) {
    if (n == 0)
        return 0;
    return square(n-1) + 2*n - 1;
}

int cube(int n) {
    if (n == 0) return 0;
    return cube(n-1) + 3*(square(n)) - 3*n + 1;
}
```

## Analisa

```
int square(int n) {
```

*Int square(int n) {* pendeklarasian fungsi square dengan menggunakan parameter formal fungsi tipe data int variabel n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int square.

```
    if (n == 0)
        return 0;
    return square(n-1) + 2*n - 1;
}
```

*if (n== 0) return 0; return square (n-1)+ 2\*n -1;* Kondisi jika nilai variabel n=0 bernilai benar maka nilai balik (return value) adalah 0 dan dioperasikan ke dalam *return square (n-1)+ 2\*n -1*. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi int square.

```
int cube(int n) {
```

*Int cube(int n) {* pendeklarasian fungsi cube dengan menggunakan parameter formal fungsi tipe data int variabel n. Bentuk umum dari pendeklarasian fungsi seperti tipe\_fungsi nama\_fungsi (parameter\_fungsi); Tanda { (kurung kurawal buka) merupakan awal dari fungsi int cube.

```
    if (n == 0) return 0;
    return cube(n-1) + 3*(square(n)) - 3*n + 1;
}
```

*if (n== 0) return 0; return cube (n-1)+ 3\*(square(n))- 3\*n +1;* Kondisi jika nilai variabel n=0 bernilai benar maka nilai balik (return value) adalah 0 dan dioperasikan ke dalam *return cube (n-1)+ 3\*(square(n))- 3\*n +1*. Tanda } (kurung kurawal tutup) merupakan akhir dari fungsi int cube.