Nama : Irfan Rahmat F

Nim : 21091397063

LAPORAN HASIL ANALISA

C. TUGAS PENDAHULUAN

Jawablah pertanyaan berikut ini :

1. Apa yang dimaksud dengan rekursi?
   Proses di mana suatu fungsi memanggil dirinya sendiri secara langsung atau tidak langsung disebut rekursi dan fungsi yang sesuai disebut sebagai fungsi rekursif. Menggunakan algoritma rekursif, masalah tertentu dapat diselesaikan dengan cukup mudah. Contoh masalah tersebut adalah Towers of Hanoi (TOH) , Inorder/Preorder/Postorder Tree Traversals , DFS of Graph , dll.

2. Tuliskan fungsi untuk menghitung nilai faktorial

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

int factorial(int n);

int main()
{
   int n;

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

   cout << "Factorial of " << n << " = " << factorial(n);

   return 0;
}

int factorial(int n)
{
   if(n > 1)
      return n * factorial(n - 1);
   else
      return 1;
}
```

3. Tuliskan fungsi untuk menampilkan nilai fibonanci dari deret fibonanci

```
#include<iostream>
```

```
using namespace std;
int fib(int n) {
  if (n <= 1)
  return n;
  return fib(n-1) + fib(n-2);
}
int main () {
  int n = 10, i;
  for(i=0;i<n;i++)
  cout<<fib(i)<<" ";
  return 0;
}
```

D. PERCOBAAN

1. 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 : Mencari nilai faktorial dari nilai yang dimasukkan oleh pengguna. Program di atas diperoleh melalui aplikasi Dev C++. Header yang saya gunakan hanya terkait dengan iostream dalam hal program I/O. Karena jenis program yang Anda tulis adalah program sekuensial, Anda harus menginisialisasi fungsi rekursif terlebih dahulu sebelum fungsi utama (). Dalam fungsi utama, pengguna memasukkan nilai dan menyimpannya dalam variabel r. Fungsi faktorial () kemudian dipanggil dengan nilai parameter di bawah nilai r dan disimpan dalam variabel hasil.

Coba perhatikan pada fungsi rekursif-nya :

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

Di sini kita membuat fungsi rekursif di mana jika nilai input adalah 1, nilai yang dikembalikan adalah nilai itu sendiri. Jika tidak, maka dihitung menggunakan rumus faktorial, yaitu (A * faktorial (A-1)).

2. 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 :

perhatikan pada fungsi rekursifnya :

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

disini kita membuat suatu guna rekursif dengan nama fibonacci dengan bawa suatu parameter ialah variabel meter. didalam guna tersebut ada percabangan if dimana bila nilai meter merupakan 0 ataupun 1 hingga nilai kembali ataupun return value pada guna tersebut merupakan nilai itu sendiri ialah 0 serta 1. Sebab algoritma deret bilangan fibonacci C++ merupakan penjumlahan dari 2 bilangan tadinya hingga kita butuh menemukan 2 nilai dini ialah 0 serta 1 supaya bisa dijumlahkan serta jadi nilai pada deret berikutnya. Apabila dijalankan program diatas hendak menciptakan output yang sama dengan program awal.

3. 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 :

Beberapa fungsi selain fungsi utama main() yaitu fungsi ambil() dan cek(). Cara kerjanya adalah pengguna akan memasukan bilangan yang ingin ditentukan apakah bilangan prima atau bukan, lau kemudian akan di cek bilangan tersebut melalui fungsi cek().Fungsi cek() berfungsi untuk mengecek apakah bilangan yang dimasukan adalah bilangan yang lebih dari 1 (karena bilangan prima dimulai dari 2) jika kondisi bernilai true maka selanjutnya akan di proses pada fungsi ambil().

Pada fungsi ambil() nilai akan dicek selama nilai i belum sama dengan 1 maka nilai bil akan terus dilakukan modulus pada nilai i dengan catatan nilai i akan ditambahkan 1 jika hasil bil%i==0 jika kondisi tidak terpenuhi ditambah 0. Fungsi cek akan menghasilkan nilai boolean (true/false) jika true maka akan menampilkan kalimat ‘Bilangan Prima’ jika tidak menampilkan kalimat ‘Bukan Bilangan Prima’.

4. 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 :

Disini bisa kita lihat pada fungsi rekursif-nya:

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

fungsi dengan nama pangkatrekursif dengan menggunakan 2 parameter yaitu nilai x dan y, jika nilai y yang dimasukan adalah 0 maka akan di set nilai baliknya adalah 1, namun jika tidak maka fungsi tersebut di set nilai baliknya dimana nilai x akan dikalikan nilai y-1.

E. 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

Code :

```
#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;
}
```

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

SUCCESSFUL (total time: 1 second)

Code :

```
include<stdio.h>
#include<stdlib.h>
#include<string.h>
int compare (const void * a, const void * b);
void allLexicographicRecur (char *str, char* data, int last, int index)
{
int i, len = strlen(str);
for ( i=0; i<len; i++ )
{
data[index] = str[i] ;
```

```
if (index == last)
printf("%s ", data);
else
allLexicographicRecur (str, data, last, index+1);
}
}
void allLexicographic(char *str)
{
int len = strlen (str) ;
char *data = (char *) malloc (sizeof(char) * (len + 1)) ;
data[len] = '\0';
qsort(str, len, sizeof(char), compare);
allLexicographicRecur (str, data, len-1, 0);
free(data);

}
int compare (const void * a, const void * b)
{
return ( *(char *)a - *(char *)b );
}
int main()
{
char str[100];
printf("Harap memasukan string yang simple,Example:ABC\n");
printf("Masukan string:");gets(str);
printf("Permutasi nya adalah: \n", str);
allLexicographic(str);
return 0;
}
```

Analisa :

Program ini berfungsi untuk mencetak urutan lexicograpich di hadapan Ulangi, program tersebut merupakan program rekursif dan dapat dilihat pada fungsi allLexicographicRecur dipanggil lagi dalam fungsi allLexicographicRecur Menjadi sebuah program, program memanggil fungsinya sendiri. Masukkan dulu. String diteruskan sebagai parameter ke fungsi allLexicographicRecur.

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

Code :

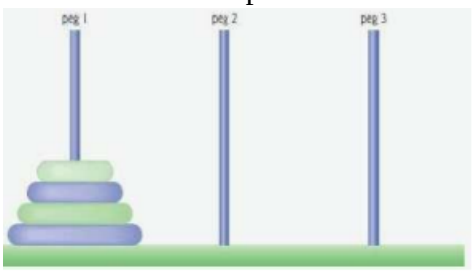
```
#include<stdio.h>
#include<stdlib.h>
#define size 10

int binsearch(int[], int, int, int);
int binsearch(int a[], int x, int low, int high) {
int mid;
   if (low > high)
   return -1;
   mid = (low + high) / 2;
   if (x == a[mid]) {
   return (mid);
 } else if (x < a[mid]) {

  binsearch(a, x, low, mid - 1);
} else {
  binsearch(a, x, mid + 1, high);
}

}

int main() {
```

```
int num, i, key, position;
int low, high, list[size];
   printf("\nMasukan Jumlah Besar data:");
   scanf("%d", &num);
   printf("\nMasukan Data:\n");

   for (i = 0; i < num; i++) {

   printf("Masukan data Index ke %d :",i);

   scanf("%d", &list[i]);
}
  low = 0;
  high = num - 1;
  printf("\nMasukan data yang ingin di cari : ");
  scanf("%d", &key);
  position = binsearch(list, key, low, high);
if (position != -1) {
  printf("\nData ke %d di temukan pada Index ke - %d", key,(position));
} else
  printf("\nData tidak di temukan");

return (0);
}
```

Analisa :

Implementasi pencarian biner, tugasnya adalah Setelah mengambil data dari array dan menentukan lokasi data, program Ini rekursif karena fungsi binsearch memanggil fungsi binsearch di dalam fungsi. cara kerja binsearch, berikan daftar, kunci, tingkat yang lebih rendah dan lebih tinggi ke fungsi binsearch kemudian rendah & lt; akan dikembalikan jika tinggi \ Jika rendah, 1 & lt; a [pertengahan] di mana rumusnya adalah mid = (rendah + tinggi) / 2. Kembali Iradiasi (a, x, rendah, sedang 1); Jika low == mid, kembalikan mid, jika tidak kembalikan mid. binsearch(a, x, sedang + 1, tinggi);

4. Buatlah program rekursif untuk memecahkan permasalahan Menara Hanoi !



Gambar 1. Menara Hanoi

Program ini merupakan program untuk menampilkan pergerakkan 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. StopCase nya 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. 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

Code :

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

void hanoi (int n, char a, char b, char c)
{
if(n==1)
{
cout<<" | Pindah Bagian atas Disc dari "<<a<<" ke "<<c<<"       |"<<endl;
}
else
{
hanoi (n-1, a, c, b);
hanoi (1 , a, b, c);
hanoi (n-1, b, a, c);
}
}

int main ()
```

```
{
int n;
cout<<" ========================================="<<endl;
cout<<" |               Program Hanoi                    |"<<endl;
cout<<" ========================================="<<endl;
cout<<" |                                                |"<<endl;
cout<<" | Masukan Jumlah Disc = ";cin>>n;
cout<<" ========================================="<<endl;
cout<<endl;
hanoi(n, 'A', 'B', 'C');
cout<<" ========================================="<<endl;
cout<<endl;

}
```

Analisa:
Program tersebut adalah implementasi Tower Haoni,Tower haoni adalah adalah sebuah permainan matematis atau teka-teki. Permainan ini terdiri dari tiga tiang dan sejumlahcakram dengan ukuran berbeda-beda yang bisa dimasukkan ke tiang mana saja. Permainandimulai dengan cakram-cakram yang tertumpuk rapi berurutan berdasarkan ukurannya dalam salah satu tiang, cakram terkecil diletakkan teratas, sehingga membentuk kerucut.
Tujuan dari teka-teki ini adalah untuk memindahkan seluruh tumpukan ke tiang yang lain, mengikuti aturan berikut:
-Hanya satu cakram yang boleh dipindahkan dalam satu waktu.
-Setiap perpindahan berupa pengambilan cakram teratas dari satu tiang dan memasukkannya ke tiang lain, di atas cakram lain yang mungkin sudah ada di tiang tersebut.
- Tidak boleh meletakkan cakram di atas cakram lain yang lebih kecil.
Pertama kita passing banyak disc dan A C B,A C B di sini adalah pasak nya,jika n sama dengan 1 maka pindahkan disc 1 dari pasak A ke pasak C,program tersebut adalah rekursi karena memanggil fungsi nya sendiri di dalam fungsinya.

5. Jelaskan proses rekursif untuk program dibawah ini !

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

Analisa : Program di atas digunakan untuk mengubah bilangan desimal menjadi bilangan biner Metode panggilan ekor rekursif. Sebuah proses yang memiliki panggilan rekursif di akhir metode dan bukan di akhir metode Ini adalah fase yang berlawanan. Untuk tidak mengonversi secara rekursif, Anda perlu mengonversi ke rekursif Parameter tambahan yang memungkinkannya dipanggil secara rekursif di akhir metode.

Deklarasikan fungsi void dengan parameter fungsi int variabel jumlah. Bentuk umum dari deklarasi fungsi adalah seperti function_type. Nama fungsi (parameter fungsi); karakter {(buka kurung kurawal) adalah awal dari fungsi voiddecTobin. jika (angka > 0).

Untuk nilai variabel num > 0, tampilannya seperti ini: Bekerja dengan deciTobin. Karakter {(tanda kurung sudut buka) adalah awal dari Untuk fungsi. Cout & lt; & lt; num% 2;}} menunjukkan hasil keluaran dari nilai variabel num modulu2. Hasil programnya. } (Kurung penutup) adalah akhir dari fungsi void. tanda } (Kurung tertutup) adalah akhir dari fungsi if. Karakter} (kurung tertutup) Akhir dari fungsi deciTobin-Void

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 : Deklarasi pencarian variabel menggunakan tipe data variabel Boolean x, size n adalah tipe data integer. Bentuk umum dari deklarasi fungsi adalah: tipe fungsi_nama fungsi (parameter fungsi); karakter {(buka kurung kurawal) Awal dari fungsi pencarian Boolean. Kondisi tidak (jika tidak) dan nilai kembalian termasuk dalam pencarian kembali dan salah. } Karakter (menutup kurung kurawal) sudah berakhir jika fungsi. } (Tutup kurung kurawal) adalah 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 : Variabel BinarySearch untuk tipe data Boolean, variabel x, Tipe data integer awal, akhir, dan n. Bentuk umum dari deklarasi fungsi Karakter {(buka kurung kurawal) seperti function_type_function_name (function_parameter); Awal dari fungsi Boolean binarySearch. Jika nilai variabel x [pertengahan] adalah n atau benar, nilai baliknya benar. {Karakter (membuka kurung kurawal) adalah awal dari Untuk fungsi. } else {if (x [mid] end, n); Jika tidak kondisi (jika tidak) Dieksekusi dalam kasus x [mid] & lt;n Kemudian nilai kembalian dimanipulasi Saat mencari secara retroaktif.

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 : Variabel tipe data bilangan bulat a dan B. Bentuk umum dari deklarasi fungsi adalah seperti function_type_function_name. (Parameter fungsi); {(Buka kurung kurawal) adalah awal dari sebuah int. fungsi rahasia. Kondisi Jika nilai variabel input b sama dengan 1, nilai is kembali memiliki nilai 0.

10. 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 :
merupakan fungsi dengan kondisi percabangan jika nilai n=0 maka nilai baliknya adalah 0, dan apabila n=1 maka nilai baliknya adalah 1, tapi jika nilai n=2 maka nilai baliknya adalah 1 dan akan dimasukkan ke operasi return 2*f(n-2) + f(n-3);.

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 :
merupakan fungsi square dengan kondisi percabangan jika n=0, maka nilai baliknya 0 dan akan dimasukkan ke rumus square(n-1) + 2*n - 1;. fungsi cube dengan kondisi percabangan jika n=0, maka nilai baliknya 0 dan akan dimasukkan ke rumus
return cube(n-1) + 3*(square(n)) - 3*n + 1.

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 :
merupakan fungsi square dengan kondisi percabangan jika n=0, maka nilai baliknya 0 dan akan dimasukkan ke rumus square(n-1) + 2*n - 1;. fungsi cube dengan kondisi percabangan jika n=0, maka nilai baliknya 0 dan akan dimasukkan ke rumus
return cube(n-1) + 3*(square(n)) - 3*n + 1.