**LAPORAN RESMI PRAKTIKUM 6**



**Dibuat Oleh :**

Nama : Putri Dwinatryska A.R.F

NIM : 21091397075

Kelas : A

**Dosen Pengampu :**

Andi Iwan Nurhidayat, S.KOM., M.T.

**UNIVERSITAS NEGERI SURABAYA**

**PROGRAM VOKASI**

**PRODI D4 MANAJEMEN INFORMATIKA TAHUN 2021**

**C. TUGAS PENDAHULUAN**

Jawablah pertanyaan berikut ini :

1. Apa yang dimaksud dengan rekursi
2. Tuluskan fungsi untuk menghitung nilai factorial
3. Tuliskan fungsi untuk menampilkan nilai fibonaci dan deret fibonaci

**JAWABAN**

1. Rekursi adalah suatu proses atau prosedur dari fungsi yang memanggil dirinya sendiri secara berulang-ulang.
2. Fungsi untuk menghitung nilai faktorial adalah

n!= n . (n-1)! Untuk n>1

0!=1 Untuk n=0 atau n=1

Secara pemrograman ditulis sebagai berikut

Faktorial (0) = 1 (1)

Faktorial (N) = N\*Faktorial( N – 1) (2)

1. Fungsi untuk menampilkan nilai fibonacci adalah

Fibonacci(1) = 1 (1)

Fibonacci(2) = 1 (2)

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

**D. PERCOBAAN**

 **Percobaan 1 : Fungsi rekursif untuk menghitung nilai faktorial**

# Input

#include <iostream> using namespace std; long int faktorial (int A); int main(){ int r,hasil;

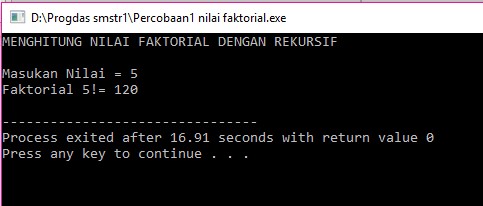
cout<<"MENGHITUNG NILAI FAKTORIAL DENGAN REKURSIF"<<endl; cout<<endl; cout<<"Masukan Nilai = "; cin>>r; hasil=faktorial(r);

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

}

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

# Output



 **Percobaan 2 : Fungsi rekursi untuk menampilkan deret fibonanci**

# Input

#include <iostream> using namespace std; int fibonacci(int m) { if (m == 0 || m ==1)

{ return m; } else {

return (fibonacci(m-1) + fibonacci(m-2));

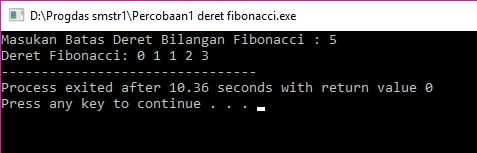
} } int main() { int n, m= 0;

cout << "Masukan Batas Deret Bilangan Fibonacci : "; cin >> n; cout << "Deret Fibonacci: "; for (int i = 1; i <= n; i++){ cout << fibonacci(m) <<" ";

m++; } return 0;

}

# Output



 **Percobaan 3 : Fungsi rekursi untuk menentukan bilangan prima atau bukan prima**

# Input

#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)){

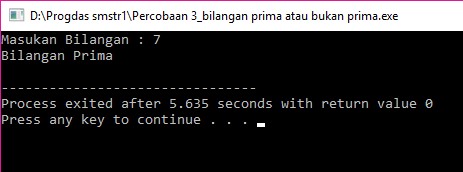
cout<<"Bilangan Prima"<<endl;

}else {

cout<<"Bukan Bilangan Prima"<<endl;

} return 0; }

# Output



 **Percobaan 4 : Fungsi rekursi untuk menghitung pangkat**

# Input

#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;

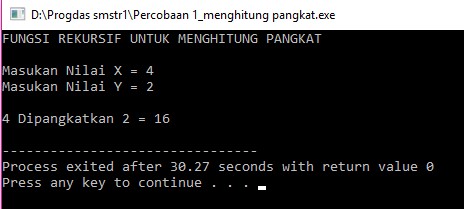
cout<<"Masukan Nilai X = "; cin>>x; cout<<"Masukan Nilai Y = "; cin>>y; cout<<endl;

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); }

# Ouput



**E. LATIHAN**

# Sorce 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<<"\tProgram rekursif menghitung segitiga Pascal \n"; cout<<"\t===========================================\n"; 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 :** Dari program tersebut dapat dilihat bahwa fungsi untuk segitiga pascal memanggil fungsinya sendiri untuk menampilkan hasil dari penjumlahan pascal.

# Sorce 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("Masukan jumlah karakter: ");gets(str); printf("Karakternyanya adalah: \n", str); allLexicographic(str); return 0; }**

**Analisa :** program di atas merupakan program rekursif untuk mencetak permutasi lexicograpich dengan adanya repitisi**.**

# Source 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("\t=====Program Binary Search=====\n"); printf("\nMasukkan Banyak data: "); scanf("%d", &num); for (i = 0; i < num; i++) { printf("Masukkan data 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 %d berada pada Indeks ke - %d", key,(position));**

**} else**

**printf("\nData tidak di temukan"); return (0); }**

**Analisa :** Program tersebut merupakan penerapan dari pencarian Binary, berfungsi untuk mencari data dalam sebuah array dan menentukan posisi data tersebut.

# Source Code

**#include <stdio.h>**

**void Hanoi(int n, char dari\_pasak, char ke\_pasak, char pasak\_tmp)**

**{**

**if (n == 1) {**

**printf("\nPindahkan disc 1 dari pasak %c ke pasak %c", dari\_pasak, ke\_pasak); return;**

**}**

**Hanoi(n-1, dari\_pasak, pasak\_tmp, ke\_pasak);**

**printf("\nPindahkan disc %d dari pasak %c ke pasak %c", n, dari\_pasak, ke\_pasak);**

**Hanoi(n-1, pasak\_tmp, ke\_pasak, dari\_pasak);**

**} int main() { int n;**

**printf("Masukan Jumlah Disk:");scanf("%d",&n);**

**Hanoi(n, 'A', 'C', 'B');**

**return 0;**

**}**

**Analisa :** jika n==1, maka pindahkan disk 1 dari pasak... ke pasak...

1. Program tersebut merupakan program konversi angka dari Desimal ke Biner. Dalam program tersebut jika angka lebih besar daripada nol (0), maka angka atau bilangan tersebut dibagi dengan nilai 2 hingga selesai. Kemudian outputnya adalah angka sisa hasil bagi dari bilangan yang telah dibagi dengan nilai 2 tersebut.
2. Program tersebut merupakan program pencarian Boolean, yaitu hanya bisa diisi dengan salah satu dari 2 pilihan nilai : **true** atau **false**. Jika ukuran yang diinput lebih besar daripada 0, dan jika (x[size-1])= n maka akan menghasilkan kondisi true. Kemudian kondisi akan menampilkan salah ketika (x, size-1, n).
3. Program tersebut merupakan program pencarian Biner Boolean. Jika akhir < awal maka salah. Nilai di tengah (mid) = (awal + akhir) / 2, jika (x[mid] == n) maka kondisi true. Jika (x[mid] < n) maka mengembalikan pencarian (x, mid+1, end, n) dan yang lain yaitu(x, start, mid-1, n).
4. Metode Misteri mengambil a dan b sebagai argumen, terdapat dua set argumen yaitu (2,25) dan (3,11). Jika (b == 0) maka kembali ke 0. Jika (b % 2 == 0), melakukan return mystery(a+a, b/2), return mystery(a+a, b/2) + a.
5. Metode Misteri dengan mengambil a dan b sebagai argumen. Jika (a ==

b) maka outputnya adalah a. m1 = (a + b) / 2, m2 = (a + b + 1) / 2. Maka mystery(a,m1), mystery(m2,b).

1. Dalam program tersebut terdapat rumus f (n). jika (n == 0) maka return 0, jika (n == 1) maka return 1, jika (n == 2) maka return 1, dan dioperasikan ke 2\*f (n-2) + f (n-3).
2. - Terdapat variabel persegi (n), jika (n == 0) maka return 0, dan dioperasikan ke persegi (n-1) + 2\*n – 1.

- Terdapat variabel kubus (n), jika (n == 0) maka return 0, dan dioperasikan ke kubus (n-1) + 3\*(square(n)) - 3\*n + 1.

**E. Penjelasan proses rekursif program ini**

Void decToBin(int num) // menjelaskan tentang fungsi konversi dari desimal ke biner

{

If (num > 0) // jika data yang disimpan pada variabel num lebih besar dari 0 akan di bagi dengan 2 dan ditampilkan sisa hasil pembagiannya

{

decToBin(num / 2);

cout << num |% 2;

}

}

6. Proses Rekursif suatu program

Boolean search(int [] x, int size, int n) { // menjelaskan tentang ukuran dari sebuah persamaan n dengan boolean.

If (size > 0) { // menjelaskan ukuran dari sebuah bilangan n.

If (x[size -1] == n) {

return true;

} else {

return search (x, size-1, n);

}

} return

false;

}

7. Proses rekrusif suatu program

boolean binarySearch {int[] x, int start, int end, int n) { // menghitung angka biner menggunakan boolean.

if (end < start)

return false;

int mid = (start+end) / 2; // hasil awal dan akhir dijumlah dan dibagi dengan 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);

8. Program rekursif suatu program

Mystery(2, 25) and mystery(3, 11)! // program mencari data nilai

Int mystery(int a, int b) {

If (b == 0);

Return 0;

If (b % 2 == 0)

Return mystery (a+a, b/2); //jumlah nilai akan dijumlah dan dibagi 2.

Return mystery (a+a, b/2) + a;

9. Program rekursif suatu program

Int mystery(int a, int b) {

If ( a == b) cout<<a<<endl;

Else {

Int m1 = (a + b) / 2; // nilai a dan b akan dijumlah dan dibagi 2

Int m2 = (a + b +1) /2; // nilai a dan b dijumlah dengan 1 lalu dibagi 2

Mystery(a, m1);

Mystery(m2, b);

}

}

10. Program rekursif suatu program

Int f (int n) { // program nilai n

If (n == 0)

Return 0;

If ( n == 1)

Return 1;

If (n == 2)

Return 1;

Return 2\*f(n-2) + f(n-3); // nilai akan dijumlahkan

11. Program rekursif suatu program

Cube(5), cube (123)? // membuat program dengan 5 nilai yang berbeda

Int square(int n) {

If (n == 0)

Return 0;

Return square (n-1) + 2\*n – 1; // nilai akan dikurang 1 ditambah 2 dan nilai lainnya akan dikurang 1.

}

Int cube(int n) {

If (in == 0) return 0;

Return cube (n-1) + 3\*square(n)) - 3\*n + 1;

}