**LAPORAN**

**PRAKTIKUM ALGORITMA DAN PEMROGRAMAN**

**MODUL 7**

****

**Disusun Oleh :**

3411201122 – Akbar Satrio Nugroho

Kelas E

20 Desember 2020

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS SAINS DAN INFORMATIKA**

**UNIVERSITAS JENDERAL ACHMAD YANI**

**BANDUNG – CIMAHI**

**2020**

**BAB I**

**HASIL PRAKTIKUM**

1. **Program Tipe Bentukan**

Tipe bentukan adalah sebuah tipe data yang tidak termasuk pada tipe data yang ada pada bahasa C ini tipe data bentukan digunakan untuk keperluan pengumpulan data yang selalu dipakai Bersama-sama atau istilahnya dalam penggunaan bahas C ini biasa digunakan syntax seperti dibawah ini:

Struct nama\_type\_record{tipe\_data nama\_variable;

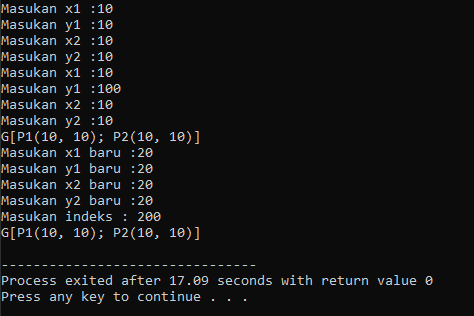
:

}var\_record

Struktur syntax tersebut resmi digunakan pada bahasa C, tipe bentukan ini bisa digunakan bersamaan dengan array.

|  |
| --- |
| #include<conio.h>  #include<stdio.h>  #define nMak 10  /\* !!! ALGORITMA !!!  type Point : < abs,ord : integer>  type Garis : < Pawal,Pakhir : Garis>  type TabGaris : < Tg[nMak+1] of Garis, nEff : integer>  \*/  //Terjemahin tipe bentukan ke algoritma ya --> Lanjutkan untuk Gaaris dan TabGaaris  //CONTOH :  typedef struct{  int abs, ord;  }Point;  typedef struct{  Point Pawal, Pakhir;  }Garis;  typedef struct{  Garis Tg[nMak+1];  int nEff;  }TabGaris;  //Prototype  void CreateTab(TabGaris \*T);  void AddElmTab(TabGaris \*T, Garis G);  void CetakTab(TabGaris T);  Garis BuatGaris(Point P1, Point P2);  Point BuatPoint(int x, int y);  void SisipIdx(TabGaris \*T, int n, Garis G);  //Main Driver  int main(){  int x1, y1, x2, y2;  TabGaris MyTab;  Garis G;  Point P1, P2;  int N;    //Memanggil Procedure CreateTab --> Untuk membuat TabGaris baru  CreateTab(&MyTab);    //Menerima masukkan x1,y1,x2,y2  printf("Masukan x1 :");scanf("%d", &x1);  printf("Masukan y1 :");scanf("%d", &y1);    printf("Masukan x2 :");scanf("%d", &x2);  printf("Masukan y2 :");scanf("%d", &y2);    //Menerima masukkan selama x1 dan y1 !=999  while(x1 == 10 && y1 == 10){  //Memanggil Procedure BuatPoint disimpan pada variable P1, dan P2  P1 = BuatPoint(x1, y1);  P2 = BuatPoint(x2, y2);    //Memanggil Procedure BuatGaris dari variabel P1, dan P2 --> Garis merupakan gabungan dari 2 buah titik  G = BuatGaris(P1, P2);    //Menambahkan elemen kedalam Tab Garis dengan cara memanggul procedure TabGaris  AddElmTab(&MyTab, G);    // Baris ini digunakan untuk menambahkan x1,y1,x2,dan y2 jika x1 dan y1 masih belum memenuhi persyaratan  printf("Masukan x1 :");scanf("%d", &x1);  printf("Masukan y1 :");scanf("%d", &y1);  printf("Masukan x2 :");scanf("%d", &x2);  printf("Masukan y2 :");scanf("%d", &y2);  }    //Pemanggilan fungsi CetakTab  CetakTab(MyTab);    //Menambahkan sebuah garis baru dengan memasukkan x1,y1,x2,y2 --> berlaku untuk menyisipkan suatu garis ke dalam tabel garis  printf("Masukan x1 baru :");scanf("%d", &x1);  printf("Masukan y1 baru :");scanf("%d", &y1);  printf("Masukan x2 baru :");scanf("%d", &x2);  printf("Masukan y2 baru :");scanf("%d", &y2);    //Memanggil kembali fungsi pembuatan point (untuk garis baru)  P1 = BuatPoint(x1, y1);  P2 = BuatPoint(x2, y2);    //Menambahkan garis baru kedalam TabGaris  G = BuatGaris(P1, P2);    //Memasukkan garis ke index tertentu  printf("Masukan indeks : ");scanf("%d", &N);    //Panggil fungsi sisipin index  SisipIdx(&MyTab, N, G); //jika pernyataannya ada bintang (termasuk output, atau input output maka menggunakan "&" dalam pemanggilannya)    //Mencetak Tabel  CetakTab(MyTab);    return 0;  }  //Realisasi Prototype  void CreateTab(TabGaris \*T){ //terjemahkan kedalam bentuk programnya ya  //(I/O T).nEff <-- 0;  (\*T).nEff = 0;  }  void AddElmTab(TabGaris \*T, Garis G){ //digunakan untuk menambahkan elm ("garis") ke dalam tab garis  //Algoritma  /\*  if((I/O T).nEff <nMak) then  (I/O T).nEff <-- (I/O T).nEff // atau T.nEff ++  (I/OT).Tg[(I/O T).nEff] <-- G;  end if  \*/    if((\*T).nEff <nMak){  (\*T).nEff++;  (\*T).Tg[(\*T).nEff] = G;  }    }  void CetakTab(TabGaris T){ //terjemahkan kedalam bentuk programnya ya  //Kamus Data  //i,x1,y1,x2,y2 : integer  int i,x1,y1,x2,y2;    //ALgoritma    /\*for (i <-- 1 to T.nEff) do  x1 <-- T.Tg[i].Pawal.abs  y1 <-- T.Tg[i].Pawal.ord  x2 <-- T.Tg[i].Pakhir.abs  y2 <-- T.Tg[i].Pakhir.ord    // output ("G[P1(%d, %d); P2(%d, %d)]\n", x1,y1,x2,y2)  endfor\*/    for (i=1;i<=T.nEff;i++){  x1 = T.Tg[i].Pawal.abs;  y1 = T.Tg[i].Pawal.ord;  x2 = T.Tg[i].Pakhir.abs;  y2 = T.Tg[i].Pakhir.ord;  printf ("G[P1(%d, %d); P2(%d, %d)]\n", x1,y1,x2,y2);  }  }  Garis BuatGaris(Point P1, Point P2){ //terjemahkan kedalam bentuk programnya ya  //Kamus Data  //G : Garis  Garis G;    //Algoritma  /\*G.Pawal <-- P1 //input  G.Pakhir <-- P2 //input\*/    G.Pawal = P1;  G.Pakhir = P2;    //--> G  return G;  }  Point BuatPoint(int x, int y){ //terjemahkan kedalam bentuk algoritmanya ya  //Kamus Data  //P : Point  Point P;    //ALgoritma  /\*P.abs <-- x //input  P.ord <-- y //input\*/    P.abs = x;  P.ord = y;    //--> P // untuk fungsi return  return P;  }  void SisipIdx(TabGaris \*T, int n, Garis G){ //Lengkapi ya adik-adik  //Kamus Data  //i, jumPindah, akhir : integer  int i, jumPindah, akhir;    //Algoritma  /\*if ((I/O T).nEff < nMak) then  jumPindah <-- (I/O T).nEff - n + 1  akhir <-- (I/O T).nEff  for (i <-- 1 to jumPindah) do  (I/O T).Tg[akhir+1] <-- (I/O T).Tg[akhir];  akhir <-- akhir - 1 //atau akhir--  end for    (I/O T).Tg(n) = G  (I/O T).nEff = (I/O T).nEff + 1 //atau (I/O T).nEff ++  end if\*/    if ((\*T).nEff < nMak){  jumPindah = (\*T).nEff - n +1;  akhir = (\*T).nEff;  for (i=1;i<=jumPindah;i++){  (\*T).Tg[akhir + 1] = (\*T).Tg[akhir];  akhir = akhir -1;  }  }  } |

1. **ScreenShot Program**



Gambar 1 ScreenShot Program Berjalan

1. **Analisa Program**

Dalam program ini ada beberapa tipe bentukan dan ada beberapa subprogram yang digunakan seperti, tipe bentukan Point, Garis dan TabGaris. Setiap tipe bentukan tersebut memiliki kegunaan yang berbeda. Perulangan while juga digunakan dalam program ini kondisinya menerima masukan selama x1 dan y1 tidak sama dengna 999 agar selama nilainya bernilai true akan terus melakukan perulangan. Pada program ini user harus menginputkan sejumlah data angka yang nantinya akan dihitung oleh program dan dicetak hasilnya pada layar.

**BAB II**

**KESIMPULAN**

Penggunaan tipe data bentukan sangat memudahkan kita apabila banyak tipe data yang akan digunakan satu sama lain dalam suatu fungsi. Tipe bentukan yang kita buat juga bisa kita beri nama sesuai dengan apa yang kita inginkan maka dari itu tipe data bentukan sangat cocok apabila kita sedang melakukan coding dengan banyak tipe data yang dibutuhkan.