PRAKTIKUM 3 #2 SDP KELAS 1 – DOSEN : SN

(Array dinamis (deklarasi dan mengisi))

- Tulis kembali naskah program berikut kemudian eksekusi!
- Beri komentar program *'secukupnya'* sehingga bila dibaca kembali akan mudah untuk dipahami!
- Tulis hasil pemahaman Anda terhadap maksud dari program-program berikut!

```
/* ------------------------------------------------------------------------------------------ */
/* File Program : PTR12.CPP                                                                   */
/* Deskripsi : Array dinamis, dimana ukuran array ditentukan dari read keyboard */
/* ------------------------------------------------------------------------------------------ */

#include <stdlib.h>
#include <stdio.h>

int main()
{
   /* Kamus Data (deklarasi, inisialisasi nilai variabel)*/
      int *tab; /* deklarasi array, perhatikan belum tampak bedanya dg
                   deklarasi pointer biasa */
      int N, i;
      /* program */
      printf("Contoh mengisi array dinamis berukuran 0..N : \n");
      printf("N = ");
      scanf("%d", &N);
      printf("Alokasi setelah mengetahui ukuran array \n");
      tab = (int *) malloc((N+1) *sizeof(int));
      if (tab != NULL)
      {
         for (i=0; i<=N; i++)
         {
            printf("i=%d tab[i]=%d \n", i, *(tab + i));
         };
         printf("Akhir program \n");
         /* dealloc */
         free(tab);  /*dealokasi*/
         return 0;
      }
      else
      {
         printf("Alokasi gagal ... \n");
         return 1;
      }
      return 0;
}
```

```
/* ------------------------------------------------------------------------------------------------------ */
/* File Program : PTR13.CPP                                                                               */
/* Deskripsi   : Array of string : Pendefinisian dan pengaksesan. Perhatikanlah permasalahannya */
/* ------------------------------------------------------------------------------------------------------ */

#include <stdlib.h>

int main()
{
   /* kamus */
   /* Definisi array yang elemennya string, statik dan sekaligus mengisi */
   static char *s[3] = {"the", "time", "is"};

   /* Definisi array yang elemennya string, dinamik */
   char *(*TabStr);     /* Deklarasi array of string */
   int i;
```

```
   /* program */
   for (i=0; i < 3; i++) {   //cetak isi s
      printf("%s\n", s[i]);
   }

   /* Alokasi TabStr sebanyak 3 */
   TabStr = (char **) malloc (3 * sizeof(char *));
   for (i=0; i < 3; i++) {
   /* Alokasi string yang merupakan elemen tabel */
      *(TabStr + i) = (char *) malloc (5 * sizeof(char));
      printf("\nInput Str[%d], maksimum 5 karakter : ", i);
      scanf("%5s", *(TabStr+i));    /* Mengisi nilai string */
      printf("\n Nilai Str[%d] : %5s\n ", i, *(TabStr + i));
   }
   return 0;
}
```

```
/* ------------------------------------------------------------------------------------------------------------ */
/* File Program : PTR14CPP                                                                                      */
/* Deskripsi   : Array of string : Pendefinisian dan pengaksesan. Perhatikanlah permasalahannya */
/* Jelaskan perbedaannya dengan program PTR13.cpp!                                                              */
/* ------------------------------------------------------------------------------------------------------------ */

#include <stdlib.h>
#define STRING char*

int main()
{
   /* kamus */
   /* Definisi array yang elemennya string, statik dan sekaligus mengisi */
   static STRING s[3] = {"the", "time", "is"};
   /* Definisi array yang elemennya string, dinamik */
   STRING (*TabStr);    /* Deklarasi array of string */
   int i;

   /* program */
   for (i=0; i < 3; i++) {  /* Print isi s */
      printf("%s\n", s[i]);
   }

   /* Alokasi TabStr sebanyak 3 */
   TabStr = (STRING *) malloc (3 * sizeof(STRING));
   for (i=0; i < 3; i++) {
   /* Alokasi string yang merupakan elemen tabel */
   *(TabStr + i) = (STRING ) malloc (5 * sizeof(char));
   printf("\nInput Str[%d], maksimum 5 karakter : ", i);
   scanf("%5s", *(TabStr+i));       /* Mengisi nilai string */
   printf("\n Nilai Str[%d] : %5s\n ", i, *(TabStr + i));
   }
   return 0;
}
```

```
/* ------------------------------------------------------------------------------------------------------------ */
/* File Program : PTR15.CPP                                                                                     */
/* Deskripsi   : Array dengan def type:mengisi dg assignment, menulis                                           */
/* ------------------------------------------------------------------------------------------------------------ */
#include <stdlib.h>

int main()
{
   /* kamus */    /* Definisi tabel integer */
   typedef struct {
      int *T;                 /* array integernya */
      int N;                  /* Ukuran efektif */
   } tabint;
   tabint MyTab;
   int i;
```

```
   /* program */
   printf("Tentukan ukuran tabel, maks 10 = ");
   scanf("%d%", &(MyTab.N));

   MyTab.T = (int *) malloc (MyTab.N * sizeof(int));

   /* isi dengan assignment */
   for (i = 0; i < MyTab.N; i++) {
      *(MyTab.T + i) = i;
      printf("i = %d MyTab.T = %d \n", i, *(MyTab.T+i));
   };
   return 0;
}
```

```
/* ------------------------------------------------------------------------------------------------------------ */
/* File Program : PTR16.CPP                                                                                     */
/* Deskripsi   : program array dinamis dan statis : mengisi dgn baca, menulis (Modular, program */
/*               passing parameter tabel/array                                                                  */
/* ------------------------------------------------------------------------------------------------------------ */

#include<stdio.h>
/* Definisi tabel integer */
typedef struct {
      int tab[10];            /* array integernya */
      int N;                  /* Ukuran efektif */
} tabint;

/* Prototype */
void incTab (tabint *T);      /* Increment setiap elemen tabel */
void printTab (tabint T);     /* Print tabel */

int main()
{
   /* kamus */
   tabint T;
   int i;

   /* program */
   T.N = 3;
   printf("Isi dan print tabel untuk indeks 1..5 \n");
   /* isi dari pembacaan */
   for (i = 0; i < T.N; i++) {
      printf("Input tabel ke -[%d] = ", i);
      scanf ("%d", &(T.tab[i]));
   };
   /* Print : perhatikan passing parameter by value */
   printTab(T);
   /* Increment : perhatikan passing parameter by reference */
   incTab (&T);
   printTab (T);
   return 0;
}

/* Body prototype */
void incTab(tabint *T)
/* Increment setiap elemen tabel */
{
   /* Kamus lokal */
   int i;

   /* Program */
   for (i = 0; i < (*T).N; i++)
   {
      (*T).tab[i] = (*T).tab[i]+1;
   }
}
```

```
void printTab(tabint T)
/* Print setiap elemen tabel */
{  /* Kamus lokal */
   int i;

   /* Program untuk traversal print */
   for (i = 0; i < T.N; i++)
   {
      printf("T[%d] = %d \n", i, T.tab[i]);
   }
}

/* ------------------------------------------------------------------------------------------------------ */
/* File Program : PTR17.CPP                                                                               */
/* Deskripsi   : Maksud program sama dengan PTR16.cpp. Cari perbedaannnya!                                */
/* ------------------------------------------------------------------------------------------------------ */

#include <stdlib.h>
/* Definisi tabel integer */
typedef struct {
      int *tab;         /* array integernya */
      int N;            /* Ukuran efektif */
} tabint;

/* Prototype */
void incTab (tabint *T);      /* Increment setiap elemen tabel */
void printTab (tabint T);     /* Print tabel */

int main()
{  /* kamus */
   tabint T;
   int i;
   /* program */
   T.tab = (int *) malloc (3*sizeof(int));
   T.N = 3;
   printf("Isi dan print tabel untuk indeks 1..3 \n");
   for (i = 0; i < T.N; i++) {      /* isi dari pembacaan */
      printf("Input tabel ke -[%d] = ", i);
      scanf ("%d", &(T.tab[i]));
   };
   printTab(T); /* Print : perhatikan passing parameter by value */
   incTab (&T); /* Increment : perhatikan passing parameter by reference */
   printTab (T);
   return 0;
}
/* Body prototype */
void incTab(tabint *T)
/* Increment setiap elemen tabel */
{   /* Kamus lokal */
   int i;
   /* Program */
   for (i = 0; i < (*T).N; i++)
   {
      (*T).tab[i] = (*T).tab[i]+1;
   }
}

void printTab(tabint T)       /* Print setiap elemen tabel */
{     /* Kamus lokal */
   int i;
   /* Program untuk traversal print */
   for (i = 0; i < T.N; i++)
   {
      printf("T[%d] = %d \n", i, T.tab[i]);
   }
}
```