Lampiran :

```
#include <conio.h>
#include <alloc.h>
#define MAX  8   /* Ukuran maksimal larik yang digunakan */


/* Deklarasi fungsi-fungsi yang digunakan */

int cektombol();
void identitas();
void terima();
void kosong();
void bantu();
void putar();
void musnah();
void isinull();

void panah_atas();
void panah_bawah();
void panah_kanan();
void panah_kiri();
void ctrl_panah_kanan();
void ctrl_panah_kiri();


/* Struktur data simpul pada senarai model double */

struct simpul
       {  char   hrf;
          struct simpul *kiri,*kanan;
       }
        *tampung[MAX+1],              /* Nama variabel larik */
        *baru,*ekor,*kepala,          /* dan pointer         */
        *tj,*hapus;


/*****  Program Utama *****/

main()
{
  clrscr();  /* Bersihkan layar */
  identitas();
  putch('A');putch('>');
  kosong();     /* Memberi nilai NULL pada elemen larik  */
  isinull();    /* Memberi nilai NULL pada semua pointer */
  terima();     /* Memanggil Prosedure terima */
}
```

```
void identitas()
{ puts("* ************************************************** *");
  puts("*            IMPLEMENTASI GABUNGAN TIPE DATA          *");
  puts("*     LARIK DIMENSI SATU DAN SENARAI MODEL DOUBLE      *");
  puts("* PADA RANCANGAN  PENAMBAHAN FASILITAS SISTEM OPERASI *");
  puts("* ------------------------------------------------- *");
  puts("*                                                    *");
  puts("*                  Di rancang oleh :                 *");
  puts("*                                                    *");
  puts("*              NAMA    :  S U M A R N O              *");
  puts("*              NIM     :  J 101 91 0550              *");
  puts("**************************************************");
  puts(" ");puts(" ");
}



int cektombol(int t)
/* Prosedure ini berfungsi untuk menyeleksi tombol-tombol *
 * yang layak untuk ditampilkan di monitor                */


{   if (t>0 && t<8 || t>8 && t<13 || t>13 && t<32 || t>127)
    return(1);
    else return (0);
}



void bantu(int *i,int *k)
{ /* Prosedure ini berfungsi untuk membuat simpul baru */

   baru=malloc(sizeof(struct simpul));     /* Pesan memori */
   if (baru==NULL)
   { puts("Memori Tidak Cukup !!!"); /* Pesan memori gagal */
     exit(1);        /* Proses eksekusi program dihentikan */
   }
   else                                  /* Pesan memori berhasil */
      { if (*i<*k) baru->hrf=tj->hrf;
        baru->kanan=NULL;
        if (kepala==NULL)
        { kepala=baru;              /* Jika belum ada simpul */
          kepala->kiri=NULL;        /* yang dibuat */
        }
        else
           { ekor->kanan=baru; /* Penggabungan simpul baru */
             baru->kiri=ekor;  /* dengan simpul lama */
           }
      }
   ekor=baru;
 }
```

```
void terima()
{ /* Prosedure ini berfungsi untuk membaca kode karakter  *
   * yang dibangkitkan oleh keyboard pada saat tombol     *
   * keyboard ditekan oleh pemakai                        */

   int i,k,x,y,tombol,tob;

   i=0;k=0;tj=NULL;
   for(;;)
   {
     do
     { do
         tombol=getch();
       while(cektombol(tombol));
       y=wherey();x=wherex();
       switch (tombol)
       { case (0) : tombol=256+getch(); /* Tombol perluasan */
                    switch(tombol)
                    { case (256+72) :gotoxy(3,y);
                                     panah_atas(&i,&x,&y);
                                     break;
                      case (256+80) :gotoxy(3,y);
                                     panah_bawah(&i,&k,&y);
                                     break;
                      case (256+77) :panah_kanan(&i,&k);
                                     break;
                      case (256+75) :panah_kiri();
                                     break;
                      case (256+115):ctrl_panah_kiri(&y);
                                     break;
                      case (256+116):ctrl_panah_kanan(&i,&y,&k);
                                     break;
                    }
                    break;
         case (8):  panah_kiri();       /* tombol backspace */
                    break;
         default :  if (tombol==13) break;
                    bantu(&i,&k);
                    baru->hrf=tombol;
                    putch(tombol);
       }
     }while (tombol!=13);     /* Jika bukan tombol <Enter> */
     gotoxy(1,y);
     putch('\n');putch('\n');putch('A');putch('>');
     if (i<k)
     { tampung[k]=kepala; /* Proses pemasukan kepala senarai */
       i=k+1;             /* ke dalam larik  */
     }
     else
        { tampung[i]=kepala;
          i++;
        }
     if (i>MAX) {putar(&k);i=MAX;} /* i sudah mencapai MAX */
     if (k<i) k=i;                 /* k merupakan pembatas */
     tj=NULL;isinull();
     }
   }
```

```
void panah_atas(int *i,int *x,int *y,int k)
{ /* Prosedure ini berfungsi untuk menangani jika tombol  *
   * yang ditekan oleh  pemakai  adalah tombol panah atas */


  if (*i==0)
  { gotoxy(*x,*y);     /* Jika indeks i menunjuk posisi */
    return;            /* elemen array teratas */
  }
  else
     { hapus=kepala;
       musnah(&k);
       isinull();
       *i=*i-1;
       tj=tampung[*i];
       gotoxy(3,*y);
       clreol();
       if (tampung[*i]==NULL) return;
       else
        { putch(tj->hrf);   /* Menampilkan isi data */
          bantu(&i,&k);       /* simpul ke satu dari  */
        }                     /* suatu senarai        */
     }
}



void panah_bawah(int *i,int *k,int *y)
{ /* Prosedure ini berfungsi untuk menangani jika tombol   *
   * yang ditekan oleh  pemakai  adalah tombol panah bawah */


  *i=*i+1;
  hapus=kepala;
  musnah(&k);
  isinull();
  gotoxy(3,*y);clreol();
  if (*i>=*k)         /* Jika indeks i melebihi pembatas k */
  { if (*i>*k) *i=*k;
    tj=NULL;
    return;
  }
  else
     { tj=tampung[*i];
       if (tampung[*i]==NULL) return;
       else
       { putch(tj->hrf);
         bantu(&i,&k);
       }
     }
}
```

```
void panah_kanan(int *i,int *k)
{ /* Prosedure ini berfungsi untuk menangani jika tombol   *
   * yang ditekan oleh  pemakai  adalah tombol panah kanan */

   if (*i>=*k) return;
   else if (tj->kanan==NULL) return;
   else if (tampung[*i]==NULL) return;
   else if (tj==NULL) tj=tampung[*i];
   else tj=tj->kanan;
   putch(tj->hrf);
   bantu(&i,&k);
}


void panah_kiri()
{ /* Prosedure ini berfungsi untuk menangani jika tombol  *
   * yang ditekan oleh  pemakai  adalah tombol panah kiri */

  int  p;
  p=wherex();
  if (p>3)
  {  putch('\b');putch(' ');
     putch('\b');     /* Hapus satu karakter di monitor */
  }

  if (ekor==kepala || ekor==NULL)
  { free(ekor);hapus=NULL;sj=NULL;
    isinull();return;
  }
  else
  {  if (tj!=NULL) tj=tj->kiri;
     hapus=ekor;          /* Menghapus satu simpul pada */
     ekor=ekor->kiri;     /* senarai lepas */
     ekor->kanan=NULL;
     free(hapus);
  }
}


void ctrl_panah_kanan(int *i,int *y,int *k)
{ /* Prosedure ini berfungsi untuk menangani jika tombol *
   * yang ditekan oleh  pemakai  adalah tombol           *
   * kontrol panah kanan                                 */

   if (*i>=*k) return;
   else if (tj->kanan==NULL) return;
   else if (tj==NULL)
   {  tj=tampung[*i];
      if (tj==NULL) return;
   }
   else tj=tj->kanan;
   if (tj==NULL)
   { gotoxy(3,*y);clreol(); }
   putch(tj->hrf);
   bantu(&i,&k);
   ctrl_panah_kanan(&i,&y,&k);    /* Secara rekursif */
}
```

```
void ctrl_panah_kiri(int *y)
{ /* Prosedure ini berfungsi untuk menangani jika tombol *
   * yang ditekan oleh  pemakai  adalah tombol          *
   * kontrol panah kiri                                 */


   if (ekor==NULL)  /* Jika senarai lepas sudah terhapus */
   { isinull();
     tj=NULL;
     gotoxy(3,*y);clreol();
     return;
   }
   panah_kiri();
   ctrl_panah_kiri(&y);  /* Pemanggilan secara rekursif */
}



void kosong()
{ /* Prosedure ini berfungsi untuk memberi nilai NULL *
   * pada semua elemen larik                           */


   register int m;
   for (m=0;m<=(MAX+1);m++)
      tampung[m]=NULL;
}



void isinull()
{ /* Prosedure ini berfungsi untuk memberi nilai NULL *
   * pada pointer-pointer yang digunakan               */

   baru=NULL;   ekor=NULL;   kepala=NULL;
}



void putar(int *k)
{ /* Prosedure ini berfungsi untuk memindahkan   *
   * elemen-elemen larik sehingga elemen ke-i    *
   * menempati elemen ke i-1                     */

   register int m;

   hapus=tampung[0];
   musnah(&k);
   tampung[0]=NULL;
   tj=&tampung[*k];
   for (m=0;m<MAX;m++)
      tampung[m]=tampung[m+1];
   tampung[MAX]=NULL;
}
```

```
void musnah(int *k)
{ /* Prosedure ini berfungsi untuk memusnahkan senarai *
   * yang tidak diperlukan lagi                        */

  if (hapus==NULL) return;
  if (hapus->kanan==NULL)
  {
    free(hapus);
    return;
  }
  else
     if (hapus==tampung[0])
     { tampung[0]=hapus->kanan;
       free(hapus);
       hapus=tampung[0];
       musnah(&k);
     }
     else
        { kepala=hapus->kanan;
          free(hapus);
          hapus=kepala;
          musnah(&k);
        }
}
```