Event.h

Type event : <

namaEvent : string

budgetKurang : integer

budgetLebih : integer

counter : integer

>

Type infotypeE : event

Type adr : pointer to elmList

Type elmList : <

Info : infotypeE

next : adr\_Event

>

Type List : <

First : adr\_Event

Last : adr\_Event

>

createListEvent (In/Out L : ListEvent)

newEvent (data : infotype) -> adr\_Event

delEvent (In/Out P : adr\_Event)

isEmptyListEvent(Out : ListEvent) -> boolean

insertFirstEvent(In/Out L : ListEvent, Out P : adr\_Event)

insertLastEvent(In/Out L : ListEvent, Out P : adr\_Event)

inserAfterEvent(In/Out L : ListEvent, Out P : adr\_Event, Out Prec : adr\_Event)

deleteFirstEvent(In/Out L : ListEvent, In/Out P : adr\_Event)

deleteLastEvent(In/Out L : ListEvent, In/Out P : adr\_Event)

deleteAfterEvent(In/Out L : ListEvent, In/Out P : adr\_Event, Out Prec : adr\_Event)

cariEvent(Out : ListEvent, namaEvent : string) -> adr\_Event

showEvent(In/Out L : ListEvent)

tambahEvent(In/Out L : ListEvent)

hapusEvent(In/Out L : ListEvent, namaEvent : string, In/Out P : adr\_Event)

Event.cpp

Procedure createListEvent (In/Out L : ListEvent)

Kamus

Algoritma

First(L) = nil

Last(L) = nil

Endprocedure

Function newEvent (data : infotype) -> adr\_Event

Kamus

P = adr\_Event

Algoritma

New ElmListEvent = P

X = info(P)

Nil = next(P)

Return P

EndProcedure

Procedure delEvent (In/Out P : adr\_Event)

Kamus

Algoritma

P delete

Endprocedure

Function isEmptyListEvent(Out : ListEvent) -> Boolean

Kamus

Algoritma

If firtst(L) = nil and last(L) = nil then

* True

Else

* False

Endif

EndProcedure

Procedure insertFirstEvent(In/Out L : ListEvent, Out P : adr\_Event)

Kamus

Algoritma

If (isEmptyListEven(L) = true then

P = First(L)

P = Last(L)

P = Next(last(L)

Else

Next(P) = first(L)

Next(last(L) = P

First(L) = P

Endif

Endprocedure

Procedure insertLastEvent(In/Out L : ListEvent, Out P : adr\_Event)

Kamus

Algoritma

If(isEmptyListEven(L) == True then

insertFirstEvent(L, P)

Else

P = next(last(L)

Next(P) = first(L)

P = last(L)

Endif

Endprocedure

Procedure inserAfterEvent(In/Out L : ListEvent, Out P : adr\_Event, Out Prec : adr\_Event)

Kamus

Algoritma

If(isEmptyListEven(L) == True then

insertFirstEven(L, P)

else

if(Prec == last(L) then

insertFirstEvent(L, P)

else if (next(Prec) == last(L) then

insertLastEvent(L, P)

else

adr\_Event Q

Q = next(Prec)

Next(Prec) = P

Next(P) = Q

Endif

Endprocedure

Procedure deleteFirstEvent(In/Out L : ListEvent, In/Out P : adr\_Event)

Kamus

Algoritma

If(isEmptyListEven(L) == True then

Output”Event Kosong”

Else if (first(L) == last(L) then

First(L) = nil

Last(L) = nil

Else

P = First(L)

First(L) = next(P)

next(P) = nil

next(last(L)) = first(L)

Endif

Endprocedure

Procedure deleteAfterEvent(In/Out L : ListEvent, In/Out P : adr\_Event, Out Prec : adr\_Event)

Kamus

Algoritma

If(isEmptyListEven(L) == True then

Output”Event Kosong”

Else if (next(first(L) == first(L) then

deleteFirstEvent(L, P)

Else

If(Prec == last(L) then

P = next(prec)

deleteFirstEvent(L, P)

Else if (next(Prec) == last(L) then

P = next(prec)

deleteLastEvent(L, P)

Else

P = next(Prec)

next(Prec) = next(P)

next(P) = nil

Endif

Endprocedure

Function cariEvent(Out : ListEvent, namaEvent : string) -> adr\_Event

Kamus

P, Q : Adr\_event

Found : boolean

Algoritma

if (isEmptyListEvent(L) == true) then

Output "Event Kosong"

Else

P = first(L)

While (P != first(L) do

If (info(P).namaEvent == namaEvent) then

Found = true

Q = P

P = next(P)

While (P != first(L)

If (found == true) then

Return Q

else if (found == false) then

return nil

endif

Endprocedure

Procedure showEvent(In/Out L : ListEvent)

Kamus

Adr\_event P = first(L)

Integer i = 1

Algoritma

If ((isEmptyListEvent(L) == true) then

Output “Event Kosong”

Else

While (P != first(L) do

Output”namaEvent”

Output”butuhBudget”

Output”budgetKurang”

Output”budgetLebih”

P = next(P)

Endif

Endprocedure

Procedure tambahEvent(In/Out L : ListEvent)

Kamus

Integer menu = 0

Event = eventBaru

Algoritma

While(menu !=2) then

While(menu ==1) do

System”CLS”

Output”Nama Event :”

Output”Dana yang dibutuhkan :”

adr\_Event P = newEvent(eventBaru)

insertFirstEvent(L, P)

Output”1. Tambah Data Event Lagi”

Output”2. Kembali”

Ouput”Pilih Menu:”

Endprocedure

Procedure hapusEvent(In/Out L : ListEvent, namaEvent : string, In/Out P : adr\_Event)

Kamus

Adr event = Q

Found : Boolean

Algoritma

If(P == nil) then

Output”Data tidak ditemukan”

Else

if P == first(L) then

deleteFirstEvent(L, P)

delEvent(P)

else if (P == last(L) then

deleteLastEvent(L, P)

delEvent(P)

else if (first(L) == last(L) then

deleteFirstEvent(L, P)

delEvent(P)

else

adr\_Event Prec

Q = first(L)

While (next(Q) != first(L) do

if (next(Q) == P) then

found = true

Prec = Q

Q = next(Q)

deleteAfterEvent(L, P, Prec);

delEvent(P);

Endif

Endprocedure

Sponsor.h

Type Sponsor : <

namaSponsor : string

budget, sisaBudget : integer

counter : integer

>

Type infotypeS : sponsor

Type adr : pointer to elmList

Type elmList : <

Info : infotypeS

Next : adr\_Sponsor

>

Type List : <

First : adr\_Sponsor

>

createListSponsor (In/Out L : ListSponsor)

newSponsor (data : infotype) -> adr\_Sponsor

delSponsor (In/Out P : adr\_Sponsor)

insertFirstSponsor(In/Out L : Sponsor, Out P : adr\_Sponsor)

insertLastSponsor(In/Out L : ListSponsor, Out P : adr\_Sponsor)

inserAfterSponsor(In/Out L : ListSponsor, Out P : adr\_Sponsor, Out Prec : adr\_Sponsor)

deleteFirstSponsor(In/Out L : ListSponsor, In/Out P : adr\_Sponsor)

deleteLastSponsor(In/Out L : ListSponsor, In/Out P : adr\_Sponsor)

deleteAfterSponsor(In/Out L : ListSponsor, In/Out P : adr\_Sponsor, Out Prec : adr\_Sponsor)

cariSponsor(Out : ListSponsor, namaSponsor : string) -> adr\_Sponsor

showSponsor(In/Out L : ListSponsor)

tambahSponsor(In/Out L : ListSponsor)

hapusSponsor(In/Out L : ListSponsor, namaSponsor : string, In/Out P : adr\_Sponsor)

Sponsor.cpp

Procedure createListSponsor (In/Out L : ListSponsor)

Kamus

Algoritma

First(L) = nil

Endprocedure

Function newSponsor (data : infotype) -> adr\_Sponsor

Kamus

P = Adr\_Sponsor

Algoritma

P = new elmListSponsor;

info(P) = x;

next(P) = nil;

return P;

Endprocedure

Procedure delSponsor (In/Out P : adr\_Sponsor)

Kamus

Algoritma

delete P

Endprocedure

Procedure insertFirstSponsor(In/Out L : Sponsor, Out P : adr\_Sponsor)

Kamus

Algoritma

If (first(L) == nil) then

first(L) = P

Else

next(P) = first(L)

first(L) = P

Endif

EndProcedure

Procedure insertLastSponsor(In/Out L : ListSponsor, Out P : adr\_Sponsor)

Kamus

Algoritma

if (first(L) == nil) then

insertFirstSponsor(L, P)

else

adr\_Sponsor Q = first(L)

while (next(Q) != nil) then

Q = next(Q)

next(Q) = P

Endif

Endprocedure

Procedure inserAfterSponsor(In/Out L : ListSponsor, Out P : adr\_Sponsor, Out Prec : adr\_Sponsor)

Kamus

Algoritma

if (first(L) == nil) then

insertFirstSponsor(L, P)

else

if (first(L) == Prec) then

insertFirstSponsor(L, P)

else if (next(Prec) == nil) then

insertLastSponsor(L, P)

else

next(P) = next(Prec)

next(Prec) = P

endif

Endprocedure

Procedure deleteFirstSponsor(In/Out L : ListSponsor, In/Out P : adr\_Sponsor)

Kamus

Algoritma

if (first(L) == nil) then

Output”List Kosong”

Else if (first(L) == nil) then

deleteFirstSponsor(L, P)

else

adr\_Sponsor Q = first(L)

while (next(next(Q)) != nil) then

Q = next(Q)

P = next(Q)

next(Q) = nil

endif

endprocedure

Procedure deleteAfterSponsor(In/Out L : ListSponsor, In/Out P : adr\_Sponsor, Out Prec : adr\_Sponsor)

Kamus

Algoritma

if (first(L) == nil) then

Output”List Kosong”

Else

if (first(L) == P) then

deleteFirstSponsor(L, P)

else if (next(P) == nil) then

deleteLastSponsor(L, P)

else

P = next(Prec)

next(Prec) = next(P)

next(P) = nil

endif

endprocedure

function cariSponsor(Out : ListSponsor, namaSponsor : string) -> adr\_Sponsor

Kamus

P = adr\_Sponsor

Algoritma

if (first(L) == nil) then

return nil

else

adr\_Sponsor Q = first(L)

bool found = false

while ((Q != nil) && (found == false) then

if (info(Q).namaSponsor == namaSponsor) then

found = true

P = Q

Q = next(Q)

if (found == true) then

return P

else

return nil

endif

endprocedure

Procedure showSponsor(In/Out L : ListSponsor)

Kamus

P = adr\_Sponsor

Integer i = 1

P = first(L)

Algoritma

If(P ==nil) then

Output”Sponsor kosong”

while (P != nil) then

Output”nama.Sponsor”

Output”budget”

Output”sisaBudget”

P = next(P)

Endif

endprocedure

Procedure tambahSponsor(In/Out L : ListSponsor)

Kamus

Integer menu = 0

Sponsor = sponsorBaru

Algoritma

while (menu != 2) do

system(“CLS”)

Output”Nama Sponsor :”

Output”Budget :”

Output”Sponsor baru budget”

insertFirstSponsor(L, newSponsor(sponsorBaru)

Output"1.Tambah Data Sponsor Lagi";

Output "2.Kembali";

Output "Pilih Menu : "

while (menu == 1)

endprocedure

Procedure hapusSponsor(In/Out L : ListSponsor, namaSponsor : string, In/Out P : adr\_Sponsor)

Kamus

Algoritma

if (P == nil) then

Output”List Kosong”

Else

if (first(L) == P) then

deleteFirstSponsor(L, P)

else if (next(P) == NULL) then

deleteLastSponsor(L, P)

else

adr\_Sponsor Prec = first(L)

while (next(Prec) != P) then

Prec = next(Prec)

deleteAfterSponsor(L, P, Prec)

endif

endprocedure

Relasi.h

Type relasi : <

jenisSponshorship : string

danaSponshorship : integer

>

Type infotypeR : relasi

Type adr : pointer to elmList

Type elmList : <

Info : infotypeR

next : adr\_Relasi

prev : adr\_Relasi

Sponsor : adr\_Sponsor

Event : adr\_Event

>

Type List : <

First : adr\_Relasi

>

createListRelasi (In/Out L : ListRelasi)

newRelasi (data : infotypeR x) -> adr\_Sponsor S, adr\_event E

delRelasi (In/Out P : adr\_Relasi)

insertFirstRelasi(In/Out L : ListRelasi, Out P : adr\_Relasi)

insertLastRelasi(In/Out L : ListRelasi, Out P : adr\_Relasi)

inserAfterRelasi(In/Out L : ListRelasi, adr\_Relasi Prec, Out P : adr\_Relasi )

deleteFirstRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi)

deleteLastRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi)

deleteAfterRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi Prec, Out Prec : adr\_Relasi)

cariRelasi(Out : ListRelasi) -> adr\_Event E, adr,\_Sponsor S

showRelasi(In/Out L : ListRelasi)

tambahRelasi(In/Out LE : ListEvent, In/Out LS : ListSponsor, In/Out LR : ListRelasi, adr\_Event E, adr\_Sponsor S, grade : string, persen : integer, )

hapusRelasi(In/Out L : ListRelasi, namaRelasi : string, In/Out P : adr\_Relasi)

Relasi.cpp

Procedure createListRelasi (In/Out L : ListRelasi)

Kamus

Algoritma

first(L) = nil

Endprocedure

Function newRelasi (data : infotypeR x) -> adr\_Sponsor S, adr\_event E

Kamus

Q = adr\_Relasi

Algoritma

Q = new elmRelasi;

Sponsor(Q) = S;

Event(Q) = E;

info(Q) = x;

next(Q) = nil;

prev(Q) = nil;

return Q;

Endprocedure

Procedure delRelasi (In/Out P : adr\_Relasi)

Kamus

Algoritma

Delete P

Endprocedure

Procedure insertFirstRelasi(In/Out L : ListRelasi, Out P : adr\_Relasi)

Kamus

Algoritma

if (first(L) == nil) then

first(L) = P

else

next(P) = first(L)

prev(first(L)) = P

first(L) = P

endif

endprocedure

Procedure insertLastRelasi(In/Out L : ListRelasi, Out P : adr\_Relasi)

Kamus

Algoritma

if (first(L) == nil) then

insertFirstRelasi(L, P)

else

adr\_Relasi Q = first(L)

while (next(Q) != nil) then

Q = next(Q)

next(Q) = P

prev(P) = Q

endif

endprocedure

Procedure inserAfterRelasi(In/Out L : ListRelasi, adr\_Relasi Prec, Out P : adr\_Relasi )

Kamus

Algoritma

if (first(L) == nil) then

insertFirstRelasi(L, P)

else

if (first(L) == Prec) then

insertFirstRelasi(L, P)

else if (next(Prec) == nil) then

insertLastRelasi(L, P)

else

next(P) = next(Prec)

prev(next(Prec)) = P

prev(P) = Prec

next(Prec) = P

endif

endprocedure

Procedure deleteFirstRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi)

Kamus

Algoritma

if (first(L) == nil) then

Output”ListKosong”

Else if (next(P) == nil) then

first(L) = nil

Sponsor(P) = nil

Event(P) = nil

Else

P = first(L)

first(L) = next(P)

prev(next(P)) = nil

next(P) = nil

Sponsor(P) = nil

Event(P) = nil

Endif

Endprocedure

Procedure deleteLastRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi)

Kamus

Algoritma

if (first(L) == nil) then

Output”ListKosong”

Else if (first(L) == P) then

deleteFirstRelasi(L,P)

else

adr\_Relasi Q = first(L);

while(next(next(Q)) != nil) then

Q = next(Q)

P = next(Q)

next(Q) = nil

prev(P) = nil

Sponsor(P) = nil

Event(P) = nil

Endif

Endprocedure

Procedure deleteAfterRelasi(In/Out L : ListRelasi, In/Out P : adr\_Relasi Prec, Out Prec : adr\_Relasi)

Kamus

Algoritma

if (first(L) == nil) then

Output”ListKosong”

Else

if (first(L) == P) then

deleteFirstRelasi(L ,P)

else if (next(P) == nil) then

deleteLastRelasi(L ,P)

else

P = next(Prec)

next(Prec) = next(P)

prev(next(P)) = Prec

next(P) = nil

prev(P) = nil

Sponsor(P) = nil

Event(P) = nil

Function cariRelasi(Out : ListRelasi) -> adr\_Event E, adr,\_Sponsor S

Kamus

adr\_Relasi Q = first(L)

Algoritma

while(Q != nil) then

if(Event(Q) == E && Sponsor(Q) == S) then

return Q

Q = next(Q)

Return nil

endprocedure

Procedure showRelasi(In/Out L : ListRelasi)

Kamus

adr\_Relasi Q = first(L)

long budget = info(Event(Q)).butuhBudget - info(Q).danaSponsorship

Algoritma

while (Q != nil) then

Output”namaEvent”

Output”namaSponsor”

Output”jenisSponshorship”

Output”budget”

Endprocedure

Procedure tambahRelasi(In/Out LE : ListEvent, In/Out LS : ListSponsor, In/Out LR : ListRelasi, adr\_Event E, adr\_Sponsor S, grade : string, persen : integer, )

Kamus

Algoritma

Procedure hapusRelasi(In/Out L : ListRelasi, namaRelasi : string, In/Out P : adr\_Relasi)

Kamus

Algoritma

if (P == nil) then

Output”ListKosong”

Else

if (first(L) == P) then

deleteFirstRelasi(L, P)

else if (next(P) == nil) then

deleteLastRelasi(L, P)

else

adr\_Relasi Prec = first(L)

while(next(Prec) != P) then

Prec = next(Prec)

deleteAfterRelasi(L, P, Prec)

endif

endprocedure

Menu.h

MainMenu (Out LE : ListEvent, Out LS: ListSponsor, Out LR: ListRelasi)

MenuEvent (In/Out LE :ListEvent, In/Out LR : ListRelasi)

MenuSponsor (In/Out LS : ListSponsor, In/Out LR : ListRelasi)

MenuRelasi(In/Out LE : ListEvent, In/Out LS : ListSponsor, In/Out LR:ListRelasi)

Spasi(jum : integer, kata : string)

Menu.cpp

Procedure MainMenu (Out LE : ListEvent, Out LS: ListSponsor, Out LR: ListRelasi)

Kamus

Menu : integer

Algoritma

Do

system("cls");

Output"||========================================================||"

Output"|| TUBES STD - Event Sponsorship ||" << endl;

Output "||========================================================||"

Output "|| Anggota : 1. Muhammad Zaidan Rafif (1302213072) ||"

Output "|| 2. Kamal Maulaazka Sidhqi (1302210032) ||"

Output "||========================================================||"

Output " MENU"

Output " 1. Event (Parent)"

Output " 2. Sponsor (Child)"

Output " 3. Level Sponsorship (relasi)"

Output " 4. Exit"

Output " Masukan menu : "

switch(menu) then

case 1 : menuEvent(LE, LR); break

case 2 : menuSponsor(LS, LR); break

case 3 : menuRelasi(LE, LS, LR); break

case 4 : exit(0); break

while (menu != 1 || menu != 2)

endprocedure

Procedure MenuEvent (In/Out LE :ListEvent, In/Out LR : ListRelasi)

Kamus

Integer : menu

Algoritma

while (menu != 3) then

Output”Daftar Event”

Output”No.”

Output”Nama Event”

Output”Dana dibutuhkan”

Output”Dana kurang”

Output”Dana Lebih”

showEvent(LE)

Output”1.Tambah Event”

Output”2. Hapus event”

Output”3. Kembali”

Output”PilihMenu”

if (menu == 1) then

ystem("CLS")

tambahEvent(LE)

else if (menu == 2) then

system("CLS");

string namaEvent;

char pil;

Output”Cari Nama Event”

adr\_Event P = cariEvent(LE, namaEvent)

if (P == nil) then

Output”Tidak ada event Bernama “

system("pause");

system("CLS");

menu = 3;

menuEvent(LE, LR)

else

Output”Nama event”

Output”Budget yang dibutuhkan”

Output”Budget Kurang”

Output”apakah anda yakin menghapus data?”

if (pil == 'y') then

adr\_Relasi R = first(LR)

while (R != nil) then

if (info(P).namaEvent == info(Event(R)).namaEvent) then

hapusRelasi(LR, R)

R = next(R)

hapusEvent(LE, namaEvent, P)

system("CLS")

menu = 3

menuEvent(LE, LR)

else if (pil == 'n') then

system("CLS")

menu = 3

menuEvent(LE, LR)

while (menu == 1 || menu == 2)

endif

endprocedure

Procedure MenuSponsor (In/Out LS : ListSponsor, In/Out LR : ListRelasi)

Kamus

Menu : integer = 0

Algoritma

while (menu != 3) then

Output”Daftar Sponsor”

Output”No.”

Output”Nama Sponsor”

Output”Budget Sponsor”

Output”Sisa Budget”

showEvent(LS)

Output”1.Tambah Sponsor”

Output”2. Hapus Sponsor”

Output”3. Kembali”

Output”PilihMenu”

if (menu == 1) then

ystem("CLS")

tambahSponsor(LS)

else if (menu == 2) then

system("CLS");

string namaSponsor;

char pil;

Output”Cari Nama Sponsor”

adr\_Event P = cariSponsor(LS, namaSponsor)

if (P == nil) then

Output”Tidak ditemukan “

system("pause");

system("CLS");

menu = 3;

menuSponsor(LS, LR)

else

Output”Nama Sponsor”

Output”Budget awal”

Output”Sisa budget”

Output”apakah anda yakin menghapus data?”

if (pil == 'y') then

adr\_Relasi R = first(LR)

while (R != nil) then

if (info(P).namaSponsor == info(Sponsor(R)).namaSponsor) then

hapusRelasi(LR, R)

R = next(R)

hapusSponsor(LS, namaSponsor, P)

system("CLS")

menu = 3

menuSponsor(LS, LR)

else if (pil == 'n') then

system("CLS")

menu = 3

menuSponsor(LS, LR)

while (menu == 1 || menu == 2)

endif

endprocedure

Procedure MenuRelasi(In/Out LE : ListEvent, In/Out LS : ListSponsor, In/Out LR:ListRelasi)

Kamus

Menu : integer = 0

char level; string namaEvent,namaSponsor

Algoritma

while (menu != 3) then

Output”Daftar Event”

Output”No.”

Output”Nama Event”

Output”Dana dibutuhkan”

Output”Dana kurang”

Output”Dana Lebih”

showEvent(LE)

Output”Daftar Sponsor”

Output”No.”

Output”Nama Sponsor”

Output”Budget sponsor”

Output”Sisa Budget”

showSponsor(LS)

Output”1.Tambah Relasi”

Output”2. List Sponsorsip”

Output”3. Kembali”

Output”PilihMenu”

while (menu == 1 || menu == 2)

endprocedure

Procedure Spasi(jum : integer, kata : string)

Kamus

Algoritma

int l = strlen(kata);

int pos = (int)((jum - l) / 2)

for (int i = 0; i < pos; i++)

Output”kata”

Endprocedure

Main.cpp

Algoritma

int main()

ListEvent LE

ListSponsor LS

ListRelasi LR

createListEvent(LE)

createListSponsor(LS)

createListRelasi(LR)

mainMenu(LE, LS, LR)

return 0

endprocedure