#define maxPilha 9

struct TpPilhaM

{

TpPilha PilhasM2

int Bases[maxPilha],

int Topos[maxPilha],

int Pilhas[maxPilha];

};

void Inicializar (TpPilhaM &PM, int N)

{

/\* PM.Topo1=-1;

PM.Topo2=maxPilha+1;\*/

int TotalP,Inicia=-1,Base=0,i;

TotalP = maxPilha/N;

PM.Topos[i]=Inicia;

PM.Bases[i]=0;

i++;

while (i<TotalP)

{

PM.Topos[i]+=Inicia+TotalP;

PM.Bases[i]+=Base+TotalP;

i++

}

PM.Bases[i]=maxPilha; //i

}

int Cheia (TpPilhaM2 &PM, int NP)

{

return PM.Base[NP+1]-1 == PM.Topo[NP];

}

int Vazia (TpPilhaM2 &PM, int NP)

{

return PM.Topo[NP] >= PM.Base[NP];

}

void Inserir (TpPilhaM2 &PM, int Elem, int NP );

{

if (!Cheia(PM, NP))

PM.Pilhas[++PM.Topo[NP]]==Elem;

}

char Retira (TpPilhaM2 &PM, int NP)

{

if (!Vazia(PM,NP))

return PM.Pilhas[PM.Topo[NP]--];

}

char RetornaElemTopo(TpPilhaM2 PM, int NP)

{

return Retira (PM,NP);

}

void Exibe (TpPilha PM, int NP)

{

while (Vazia(PM,NP))

printf ("%c"-Retira(PM,NP));

}

#define maxPilha 10

struct TpPilhaM

{

int Topo1, Topo2;

char Pilhas[maxPilhas];

};

void Inicializar (TpPilhaM &PM)

{

PM.Topo1=-1;

PM.Topo2=maxPilha+1;

}

void Inserir (TpPilhaM &PM, char Elem, int NP )

{

if(NP==1)

PM.Pilhas[++PM.Topo1]=Elem;

else

PM.Pilhas[--PM.Topo2]=Elem;

}

int Cheia (int Topo1, int Topo2)

{

return Topo1+1 == Topo2 ;

}

char Retira (TpPilhaM &PM, int NP);

{

if (NP==1)

return PM[P.Topo1--];

else

return PM[P.Topo++];

}

int Vazia (int Topo, int NP)

{

if (NP==1)

return Topo==-1;

else

return Topo==maxPilha;

}

char RetornaElemento (TpPilhaM PM, int NP)

{

if (NP==1)

return PM.Pilhas[PM.Topo1];

else

return PM.Pilhas[PM.Topo2];

}

void Exibe (TpPilha PM, int NP)

{

if (NP==1)

while (!Vazia(PM.Topo1,NP))

printf ("\n %c",Retira(PM,NP));

else

while(!Vazia(PM.Topo2,NP))

printf ("\n %c",Retira(PM,NP));

}