/\*ASSIGNMENT-1

Name:SENTHIL KUMAR S A

USN:1PE17CS136

Class: B.E. 3rd SEMESTER, CSE

Section:'B'

Question no-1

Program to allocate memory dynamically to a 1D array.\*/

#include<stdio.h>

#include<stdlib.h>

int main()

{

int n;

printf("Enter no of elements in 1D array: ");

scanf("%d",&n);

int \*p=(int \*)malloc(n\*sizeof(int)),i;

printf("Enter array elements: ");

for(i=0;i<n;i++)

scanf("%d",p+i);

printf("The array elements are: ");

for(i=0;i<n;i++)

printf("%d ",\*(p+i));

printf("\n");

free(p);

return 0;

}

Output

Enter no of elements in 1D array: 7

Enter array elements: 1 3 5 7 2 4 6

The array elements are: 1 3 5 7 2 4 6

Process returned 0 (0x0) execution time : 10.045 s

Press any key to continue.

/\*Question no -2

Program to allocate memory dynamically to 2D Arrays.\*/

#include<stdio.h>

#include<stdlib.h>//for exit(),malloc() and free()

int main()

{

int \*\*a,row,col,i,j ;

printf("Enter no of rows and columns: ");

scanf("%d%d",&row,&col);

a=(int\*\*)malloc(row\*sizeof(int \*));

if(a==NULL)

{

printf("No memory\n");

exit(0);

}

for(i=0;i<row;i++)

a[i]=(int\*)malloc(col\*sizeof(int));

printf("Enter the elements: ");

for(i=0;i<row;i++)

for(j=0;j<col;j++)

scanf("%d",(\*(a+i)+j));

printf("The array is: ");

for(i=0;i<row;i++)

{

printf("\n");

for(j=0;j<col;j++)

printf("%d ",\*(\*(a+i)+j));

}

printf("\n");

free(a)

return 0;

}

Output

Enter no of rows and columns: 4 4

Enter the elements: 1 3 5 7 9 2 4 6 8 1 3 5 7 9 2 4

The array is:

1 3 5 7

9 2 4 6

8 1 3 5

7 9 2 4

Process returned 0 (0x0) execution time :12.724 s

Press any key to continue.

/\*Question no-3

Program to create a self-referential structure.Read in values for four structure variables and print them. \*/

#include<stdio.h>

#include<stdlib.h>

typedef struct N

{

int data;

struct N \*ptr ;

}list ;

int main()

{

list l1,l2,l3,l4;

l1.ptr=l2.ptr=l3.ptr=l4.ptr=NULL ;

int d,e,f,g;

printf("Enter data value for l1, l2, l3, l4: ");

scanf("%d%d%d%d",&d,&e,&f,&g);

l1.data=d;

l2.data=e;

l3.data=f;

l4.data=g;

l1.ptr=&l2;

l2.ptr=&l3;

l3.ptr=&l4;

printf("l1:\nData:%d\nLink:%u\nl2:\nData:%d\nLink:%u ",l1.data,l1.ptr,l2.data,l2.ptr);

printf("\nl3:\nData:%d\nLink:%u\nl4:\nData:%d\nlink:%d\n",l3.data,l3.ptr,l4.data,l4.ptr);

return 0;

}

Output

Enter data value for l1, l2, l3, l4: 12 14 13 15

l1:

Data:12

Link:6356736

l2:

Data:14

Link:6356728

l3:

Data:13

Link:6356720

l4:

Data:15

link:0

Process returned 0 (0x0) execution time : 12.375 s

Press any key to continue.

/\*Question no-4

Program to declare variables of structure type and print data using pointer-to-structure variable.\*/

#include<stdio.h>

#include<stdlib.h>

typedef struct n

{

int data;

struct n \*link;

}list;

int main()

{

list l1,l2,l3,\*p ;

printf("Enter three data values for three nodes: ");

scanf("%d%d%d",&l1.data,&l2.data,&l3.data);

l1.link=&l2;

l2.link=&l3;

l3.link=NULL;

printf("The linked list is: ");

p=&l1;

while(p!=NULL)

{

printf("%d->",p->data);

p=p->link;

}

printf("end\n");

return 0;

}

Output

Enter three data values for three nodes: 13 03 99

The linked list is: 13->3->99->end

Process returned 0 (0x0) execution time: 18.919 s

Press any key to continue.

/\*Question-5

Multiplying two polynomials \*/

#include<stdio.h>

#define MAX 100

typedef struct

{

int expo ;

float coef;

}poly ;

int pmult(poly a[],int m,poly b[],int n,poly c[]);//polynomial multiplication

void display(poly a[],int m);//display polynomial

void input(poly a[],int m);//input polynomial

void sort(poly a[],int m);//sort the polynomial in decreasing order of exponents

int main()

{

int m,n ;

poly a[MAX],b[MAX],c[MAX] ;

printf("Enter no of terms in 1st polynomial: ");

scanf("%d",&m);

printf("Enter terms of polynomial A: ");

input(a,m);

printf("Enter no of terms in 2nd polynomial: ");

scanf("%d",&n);

printf("Enter terms of polynomial B: ");

input(b,n);

printf("POLYNOMIAL A: ");

display(a,m);

printf("\nPOLYNOMIAL B: ");

display(b,n);

int size=pmult(a,m,b,n,c);

printf("\nThe product is: ");

display(c,size);

printf("\n");

return 0;

}

int pmult(poly a[],int m,poly b[],int n,poly c[])

{

poly temp[MAX];

int i,j,tIndex=0 ;

//multiplication

for(i=0;i<m;i++)

{

for(j=0;j<n;j++)

{

temp[tIndex].coef=a[i].coef \* b[j].coef;

temp[tIndex].expo=a[i].expo + b[j].expo;

tIndex++;

}

}

temp[tIndex].expo=-1;

//processing to reduce redundant terms

int cIndex=0;

for(i=temp[0].expo;i>=temp[tIndex-1].expo;i--)

{

c[cIndex].coef=0;

for(j=0;j<tIndex;j++)

{

if(temp[j].expo==i)

{

c[cIndex].coef+=temp[j].coef;

c[cIndex].expo=temp[j].expo ;

}

}

cIndex++;

}

return cIndex;

}

void input(poly a[],int m)

{

int i;

printf("\n");

for(i=0;i<m;i++)

{

printf("Enter exponent and coefficient of term no %d: ",i+1);

scanf("%d%f",&a[i].expo,&a[i].coef);

}

sort(a,m);

}

void display(poly a[],int m)

{

int i;

printf("\n");

for(i=0;i<m;i++)

{

printf("%0.2fx^(%d)",a[i].coef,a[i].expo);

if(i!=m-1)

printf("+");

}

}

void sort(poly a[],int m)

{

int i,j ;

int large,pos,temp;

float tmp;

for(i=0;i<m-1;i++)

{

large=a[i].expo;

pos=i;

for(j=i+1;j<m;j++)

{

if(a[j].expo>large)

{

large=a[j].expo;

pos=j;

}

}

temp=a[pos].expo;

a[pos].expo=a[i].expo;

a[i].expo=temp;

tmp=a[pos].coef;

a[pos].coef=a[i].coef;

a[i].coef=tmp ;

}

}

Output

Enter no of terms in 1st polynomial: 4

Enter terms of polynomial A:

Enter exponent and coefficient of term no 1: 0 2

Enter exponent and coefficient of term no 2: 1 3

Enter exponent and coefficient of term no 3: 2 4

Enter exponent and coefficient of term no 4: 3 5

Enter no of terms in 2nd polynomial: 4

Enter terms of polynomial B:

Enter exponent and coefficient of term no 1: 3 5

Enter exponent and coefficient of term no 2: 2 4

Enter exponent and coefficient of term no 3: 1 3

Enter exponent and coefficient of term no 4: 0 2

POLYNOMIAL A:

5.00x^(3)+4.00x^(2)+3.00x^(1)+2.00x^(0)

POLYNOMIAL B:

5.00x^(3)+4.00x^(2)+3.00x^(1)+2.00x^(0)

The product is:

25.00x^(6)+40.00x^(5)+46.00x^(4)+44.00x^(3)+25.00x^(2)+12.00x^(1)+4.00x^(0)

Process returned 0 (0x0) execution time : 32.702 s

Press any key to continue.

/\*Question no-6

Program to perform binary search on a sorted array iteratively.\*/

#include<stdio.h>

#define MAX 100

void bsort(int a[],int n);

int bsearch(int a[],int n,int ele);

int main()

{

int arr[MAX],n,i,item ;

printf("Enter no of elements in array: ");

scanf("%d",&n);

printf("Enter array elements: ");

for(i=0;i<n;i++)

scanf("%d",arr+i);

bsort(arr,n);

printf("Enter element to be searched in array: ");

scanf("%d",&item);

int res=bsearch(arr,n,item);

if(res==-1)

printf("Element not found\n");

else

printf("Found at position:%d\n",res);

return 0;

}

void bsort(int a[],int n)

{

int i,j,temp;

for(i=0;i<n-1;i++)

{

for(j=0;j<n-i-1;j++)

{

if(a[j]>a[j+1])

{

temp=a[j];

a[j]=a[j+1];

a[j+1]=temp;

}

}

}

}

int bsearch(int a[],int n,int ele)

{

int mid,beg=0,end=n-1;

while(beg<=end)

{

mid=(beg+end)/2;

if(ele==a[mid])

return mid;

else if(ele>a[mid])

beg=mid+1;

else

end=mid-1;

}

return -1;

}

Output

Enter no of elements in array: 7

Enter array elements: 1 3 5 6 7 8 9

Enter element to be searched in array: 8

Found at position:5

Process returned 0 (0x0) execution time : 22.694 s

Press any key to continue.

/\*Question no-7

Program to perform binary search on a sorted array recursively\*/

#include<stdio.h>

#define MAX 100

void bsort(int a[],int n);

int bsearch(int a[],int n,int ele,int,int);

int main()

{

int arr[MAX],n,i,item ;

printf("Enter no of elements in array: ");

scanf("%d",&n);

printf("Enter array elements: ");

for(i=0;i<n;i++)

scanf("%d",arr+i);

bsort(arr,n);

int beg=0,end=n-1;

printf("Enter element to be searched in array: ");

scanf("%d",&item);

int res=bsearch(arr,n,item,beg,end);

if(res==-1)

printf("Element not found\n");

else

printf("Found at position:%d\n",res);

return 0;

}

void bsort(int a[],int n)

{

int i,j,temp;

for(i=0;i<n-1;i++)

{

for(j=0;j<n-i-1;j++)

{

if(a[j]>a[j+1])

{

temp=a[j];

a[j]=a[j+1];

a[j+1]=temp;

}

}

}

}

int bsearch(int a[],int n,int ele,int beg,int end)

{

int mid=(beg+end)/2;

if(beg<=end)

{

mid=(beg+end)/2;

if(ele==a[mid])

return mid;

else if(ele>a[mid])

return bsearch(a,n,ele,mid+1,end);

else

return bsearch(a,n,ele,beg,mid-1);

}

return -1;

}

Output

Enter no of elements in array: 7

Enter array elements: 1 3 5 6 7 8 9

Enter element to be searched in array: 1

Found at position:0

Process returned 0 (0x0) execution time: 9.768 s

Press any key to continue.