**Data Structures and Algorithm**

**Customize Implementation of Array Data Structure**

**-------------------------------------------------------------------------**

**“**

**Developed By: Rushikesh Kadu**

**Implementation: Customize ArrayADT using C**

**“**

**-------------------------------------------------------------------------**

ArrayADT.c

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

struct ArrayADT

{

int capacity;

int lastindex;

int \*ptr;

};

struct ArrayADT\* createArray(int cap)

{

struct ArrayADT \*arr = (struct ArrayADT\*)malloc(sizeof(struct ArrayADT));

arr->capacity = cap;

arr->lastindex = -1;

arr->ptr = (int\*)malloc(sizeof(int)\*cap);

return arr;

}

void append(struct ArrayADT \*arr,int data)

{

if(arr->lastindex==arr->capacity-1)

printf("Overflow");

else if(arr->lastindex<arr->capacity-1)

{

arr->lastindex+=1;

arr->ptr[arr->lastindex] = data;

}

}

void insert(struct ArrayADT \*arr,int index,int data)

{

if(arr->lastindex==arr->capacity-1)

printf("Overflow");

else{

int i;

if(index>=0 && index<=arr->lastindex+1)

{

for(i=arr->lastindex+1;i>index;i--)

arr->ptr[i] = arr->ptr[i-1];

arr->ptr[i] = data;

arr->lastindex+=1;

}

}

}

void removeElement(struct ArrayADT \*arr,int index)

{

if(arr->lastindex==-1)

printf("Underflow");

else

{

if(index<=arr->lastindex)

{

int i;

for(i=index;i<arr->lastindex;i++)

arr->ptr[i]=arr->ptr[i+1];

arr->lastindex-=1;

}

}

}

int getItem(struct ArrayADT \*arr,int index)

{

if(index>=0 && index<=arr->lastindex)

return arr->ptr[index];

}

int searchItem(struct ArrayADT \*arr,int data)

{

int i;

for(i=0;i<=arr->lastindex;i++)

if(arr->ptr[i]==data)

return i;

return -1;

}

void release(struct ArrayADT \*arr)

{

free(arr->ptr);

free(arr);

}

int count(struct ArrayADT \*arr)

{

return arr->lastindex+1;

}

void editItem(struct ArrayADT \*arr,int index,int data)

{

if(index>=0 && index<=arr->lastindex)

arr->ptr[index] = data;

}

void display(struct ArrayADT \*arr)

{

if(arr==NULL || arr->lastindex==-1)

printf("ArrayADT is Empty");

else

{

int i;

for(i=0;i<=arr->lastindex;i++)

printf("%d ",arr->ptr[i]);

}

printf("\n");

}

int menu()

{

int choice;

printf("\n1.CreateArray()");

printf("\n2.Append Element()");

printf("\n3.Insert Element()");

printf("\n4.Remove Element()");

printf("\n5.Release Memory of array");

printf("\n6.Count Element()");

printf("\n7.GetItem()");

printf("\n8.SearchItem()");

printf("\n9.EditItem()");

printf("\n10.Exit()");

printf("\nEnter Your Choice:");

scanf("%d",&choice);

return choice;

}

void main()

{

struct ArrayADT \*arr = NULL;

int size,data,index,c=-1;

while(1)

{

system("cls");

if(c!=-1)

{

printf("%d ",c);

c=-1;

}

else

display(arr);

switch(menu())

{

case 1:

printf("\nEnter Size to Create Array:");

scanf("%d",&size);

arr = createArray(size);

break;

case 2:

printf("\nEnter Data:");

scanf("%d",&data);

append(arr,data);

break;

case 3:

printf("\nEnter Index:");

scanf("%d",&index);

printf("Enter Data to Insert:");

scanf("%d",&data);

insert(arr,index,data);

break;

case 4:

printf("Enter Index to remove Element:");

scanf("%d",&index);

removeElement(arr,index);

break;

case 5:

release(arr);

break;

case 6:

c = count(arr);

break;

case 7:

printf("Enter Index to get Element:");

scanf("%d",&index);

c = getItem(arr,index);

break;

case 8:

printf("Enter Data:");

scanf("%d",&data);

c = searchItem(arr,data);

break;

case 9:

printf("Enter Index to edit Data:");

scanf("%d",&index);

printf("Enter Data:");

scanf("%d",&data);

editItem(arr,index,data);

break;

case 10:

exit(1);

}

}

}

Output :-

