EE4013 Assignment-1

PEDAVEGI ADITYA - EE18BTECH11034

Download all the codes from

https://github.com/adi2000pedavegi/EE4013/tree/main/Assignment−1/codes

and latex-tikz codes from

https://github.com/adi2000pedavegi/EE4013/tree/main/Assignment−1/figs

1 Problem

Consider the following C program

```
#include <stdio.h>
int main ()
{
        int arr[] = {1,2,3,4,5,6,7,8,9,0,1,2,5}, *ip
            = arr + 4;
        printf("%d\n",ip[1]);
        return 0;
}
```

The number that will be dispalyed on the execution of the program is :

2 Solution

The output of the given C program is

```
6
```

Initially we defined an integer array by name *arr* of size 13.

```
int arr[] = {1,2,3,4,5,6,7,8,9,0,1,2,5};
```

An array in C or be it in any programming language is a collection of similar data items stored at contiguous memory locations and elements can be accessed randomly using indices of an array. They can be used to store collection of primitive data types such as int, float, double, char, etc of any particular type.
The created array will be created as shown in Fig.0

Given Array

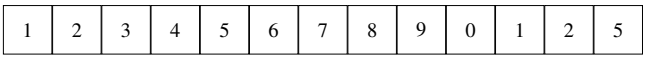

Fig. 0: Given Input array

By default while creating an array, a pointer as the name of the array i.e., *arr* is created and it stores the address of the first element in the arr.
This is shown in below Fig.0
Then we initialized a pointer ip as

```
int *ip = arr + 4;
```

The above line of code creates a pointer such that it points to the 4^{th} element from the element pointed by pointer *arr* i.e., the pointer ip stores the address of the element present in 5^{th} index of array arr. This is pictured in Fig.0

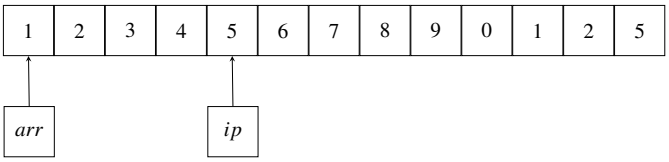

Fig. 0: Pointer pointing to an element in array

Now the final line of code is:

```
printf("%d\n",ip[1]);
```

Using this line of code we are printing the element present in $ip[1]$. The value of $ip[1]$ i.e $(*(ip+1))$ is equivalent to element present in memory address pointed by the pointer $ip+1$.
The element pointed by pointer $ip+1$ is shown in Fig. 0
So, now the above printf line prints the integer stored at the address, the pointer $ip+1$ points in the terminal i.e., **6**.

3 Vector Form Approach

Consider the given array as a vector A, multiplying it with another vector of same length having 0's

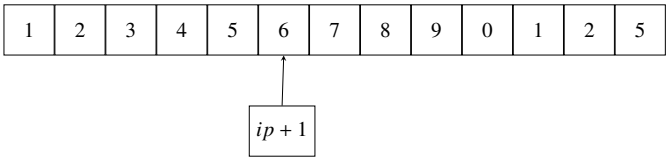

Fig. 0

and 1's. The other vector have 0's at all indexes except at the required index.
Consider the identity matrix I of size *nxn*

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & . & . & . & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & . & . & . & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & . & . & . & 0 & 0 & 0 \\ . & . & . & . & . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & . & . & . & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & . & . & . & 0 & 0 & 1 \end{bmatrix}$$

The eigen vectors of the above I matrix gives n unit vectors e_0 to e_{n-1} each of size $nx1$ where

$$e_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix} e_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ . \\ . \\ 0 \end{bmatrix} .....e_{n-1} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ 1 \end{bmatrix}$$

Hence the column unit vector e_i has value 1 at ith place (index) and 0's at other indexes.
The given pointer ip gives the 4th index in arr.So the value $ip[1]$ i.e $(^*(ip+1))$ gives the 5th index in arr i.e **arr[5]**
Element wise multiplication (inner product) of two vectors A and e_5 gives the required output.

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 0 \\ 1 \\ 2 \\ 5 \end{bmatrix}$$

$$e_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Vector multiplication of A and e_5 i.e $A^T e_5$ gives

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 & 2 & 5 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 6 \end{bmatrix}$$