Please try to run given code and try to make sense of the output. Fiddle with the code and see changes in the output. Before you start coding for rest of the questions in this lab, it is important that you understand given code.

Check "Test against custom input" box before clicking "Run Code" button.

Input Format

-

Constraints

-

Output Format

-

Sample Input 0

```
Aman 1 9.5
Raman 2 9.05
Karan 3 9.30
Manan 4 9.7
Kanan 5 9.9
```

Sample Output 0

```
Aman 1 9.500000
Aman 1 9.500000
Aman 1 9.500000
Aman 1 9.500000
Aman 1 9.500000
Raman 2 9.050000
Raman 2 9.050000
Raman 2 9.050000
Raman 2 9.050000
Raman 2 9.050000
Karan 3 9.300000
Karan 3 9.300000
Karan 3 9.300000
Karan 3 9.300000
Karan 3 9.300000
Manan 4 9.700000
Manan 4 9.700000
Manan 4 9.700000
Manan 4 9.700000
Manan 4 9.700000
Kanan 5 9.900000
Kanan 5 9.900000
Kanan 5 9.900000
Kanan 5 9.900000
Kanan 5 9.900000
```

Submissions:

52

Max Score:

10

Difficulty:

Easy

Rate This Challenge:

Download problem statement

Download all test cases

Suggest Edits

Collapse

Admin Options

Edit Challenge
View Submissions

C++20

Accessing dynamic array in different ways

```
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

typedef struct
{
   int rno;
   float cpi;
   char nm[13];
}stu;

int main()
{
   int i=0;

   void *v=malloc(5*sizeof(stu));

   stu *e=(stu *)v;
   stu *e1=(stu *)v;

   stu (*t) [5]=(stu (*) [5])v;

   for(i=0;i<5;i++)
   {
      cin>> (e+i)->nm >> (e+i)->rno >> (e+i)->cpi;
   }

   for(i=0;i<5;i++)
   {
      cout<< (e+i)->nm << " "<<(e+i)->rno << " "<<(e+i)-
>cpi<<endl; // Pointer and Offset

      cout<<(&e[i])->nm<<" "<<(&e[i])->rno<<" "<<(&e[i])-
>cpi<<endl; // Index Notation
```

```
        cout<< (e[i]).nm <<" "<< (e[i]).rno <<" "<<
(e[i]).cpi<<endl; // Index Notation

        cout<< (e1)->nm <<" "<< (e1)->rno<<" " << (e1)-
>cpi<<endl; // Pointer Incrementation
        e1++;

        //Index Notation with Pointer of Memory
        cout<<  (&t[0][i])->nm <<" "<< (&t[0][i])->rno <<" "
<<(&t[0][i])->cpi<<endl;

        cout<<endl;
    }

    free(t);

    return 0;
}Accessing dynamic array in different ways
```