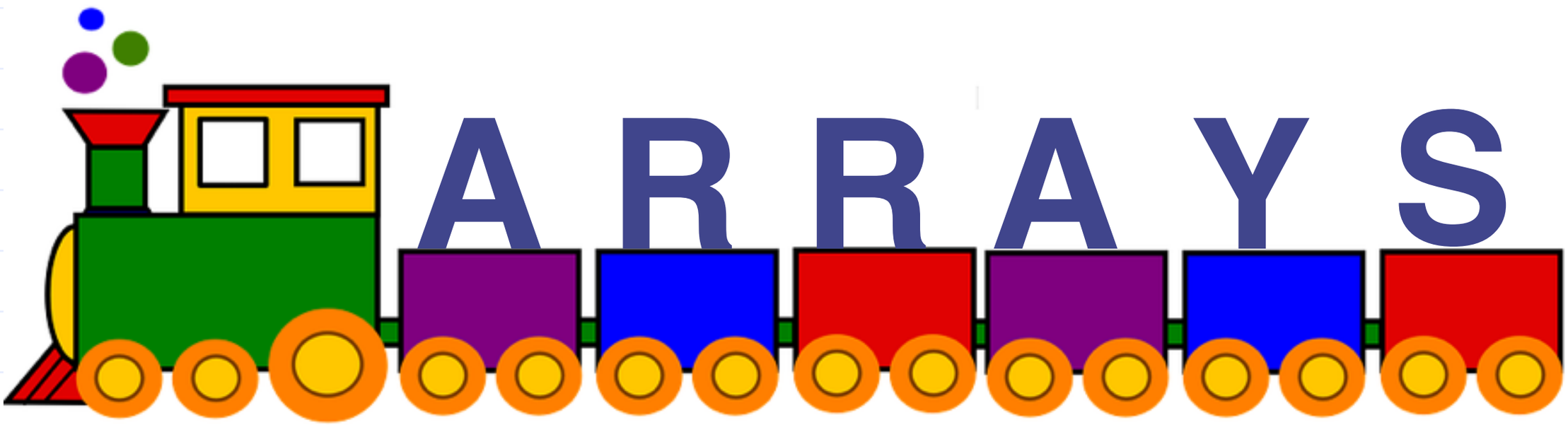
ESC101: Introduction to Computing

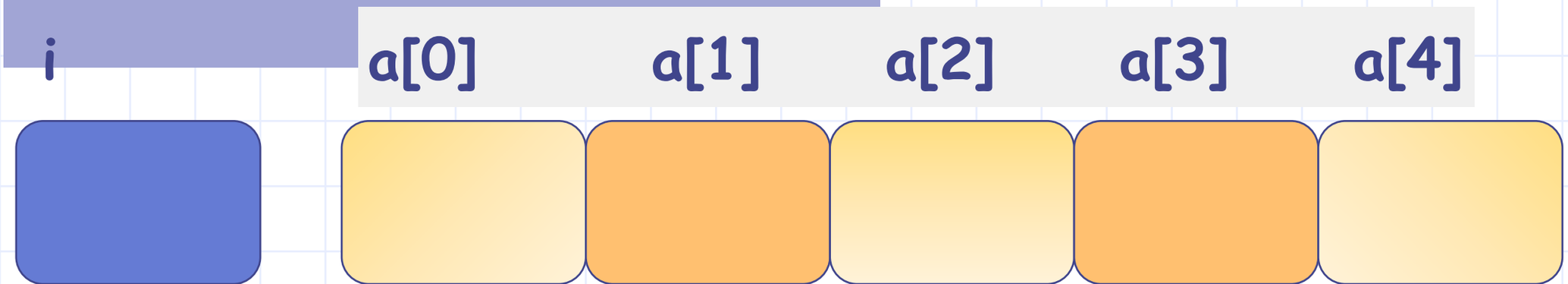


```
include <stdio.h>
int main () {
      int i;
      int a[5];

       for (i=0; i < 5; i= i+1) {
            a[i] = i+1;

            printf("%d", a[i] );
       }
       return 0;
}
```

The program defines an integer variable called i and an integer array with name a of size 5

This is the notation used to address the elements of the array.

- The variable i is being used as an "index" for a.
- Similar to the math notation a_i



Practice Problem

- Write a program to read in an array of 5 integers. Compute and print the running total of the integers.

- Input: 3 1 5 2 9
- Output: 3 4 9 11 20

Solution for Practice problem

```
#include <stdio.h>
int main()
{
   int arr[5];
   //read input
   for(int i=0; i<5; i++)
      scanf(“%d",&arr[i]);

   //compute running total
   for(int i=1; i<5; i++)
      arr[i] = arr[i-1]+arr[i];

   //obtain output
   for(int i=0; i<5; i++)
      printf(“%3d",arr[i]);

   printf("\n");
   return 0;

}
```

Practice Problem

- Write a program to read in an array of 5 integers and do a one element rotation of the input array.

- Input: 3 1 5 2 9
- Output: 1 5 2 9 3

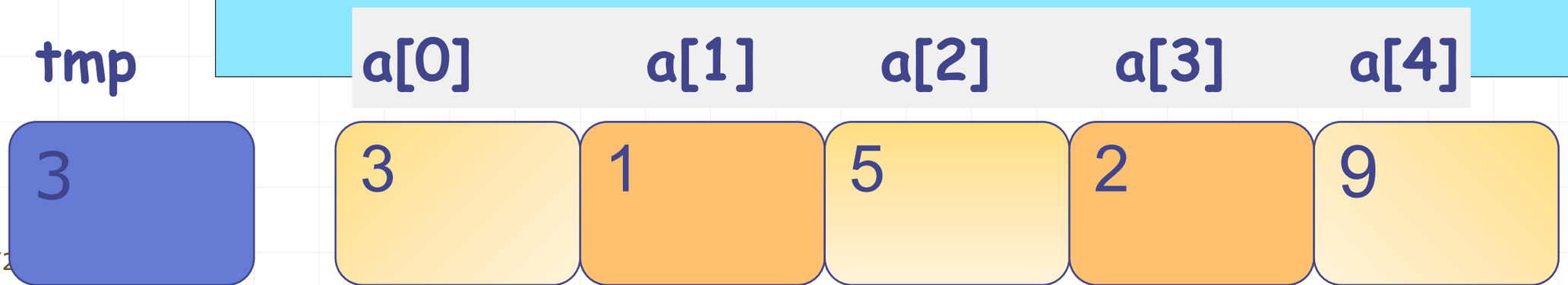
Solution for Practice problem

```
#include <stdio.h>
int main()
{
   int arr[5], tmp;

   for(int i=0; i<5; i++)
      scanf(“%d",&arr[i]);


   for(int i=0;i<5;i++)
      printf(“%d ",arr[i]);
   printf("\n");
   return 0;
}
```

i	a[0]	a[1]	a[2]	a[3]	a[4]
0	3	1	5	2	9

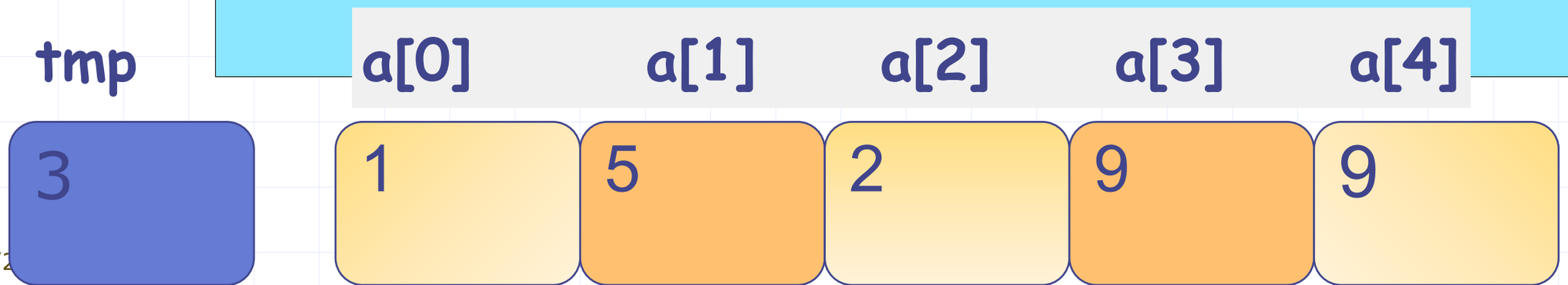
Solution for Practice problem

```
#include <stdio.h>
int main()
{
   int arr[5], tmp;

   for(int i=0; i<5; i++)
      scanf(“%d",&arr[i]);

   tmp =arr[0];

   for(int i=0;i<5;i++)
      printf(“%d ",arr[i]);
   printf("\n");
   return 0;
}
```



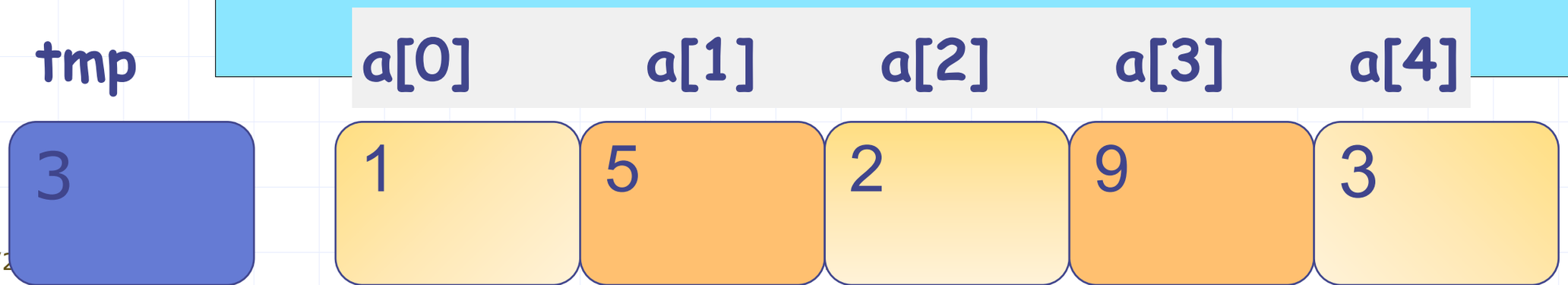
Solution for Practice problem

```
#include <stdio.h>
int main()
{
   int arr[5], tmp;

   for(int i=0; i<5; i++)
      scanf(“%d",&arr[i]);

   tmp =arr[0];
   for(int i=1; i<5; i++)
      arr[i-1]=arr[i];

   for(int i=0;i<5;i++)
      printf(“%d ",arr[i]);
   printf("\n");
   return 0;
}
```

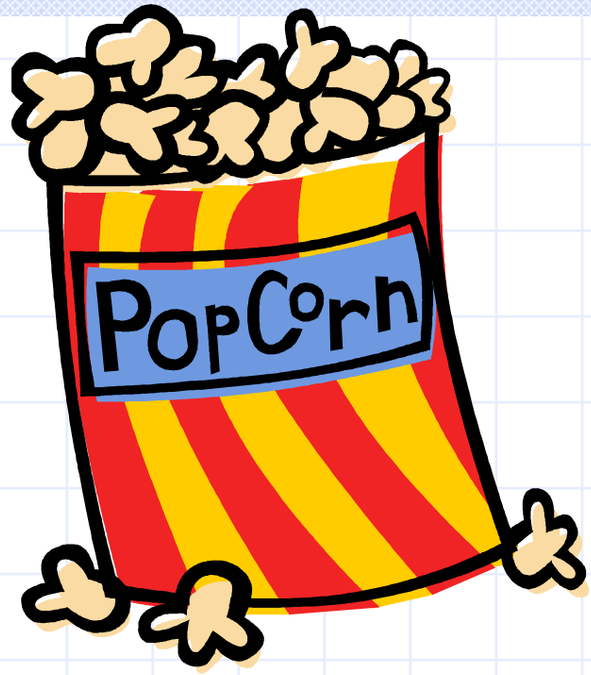


Solution for Practice problem

```
#include <stdio.h>
int main()
{
   int arr[5], tmp;

   for(int i=0; i<5; i++)
      scanf(“%d",&arr[i]);

   tmp =arr[0];
   for(int i=1; i<5; i++)
      arr[i-1]=arr[i];
   arr[4]=tmp;

   for(int i=0;i<5;i++)
      printf(“%d ",arr[i]);
   printf("\n");
   return 0;
}
```



Mind the size(of array)

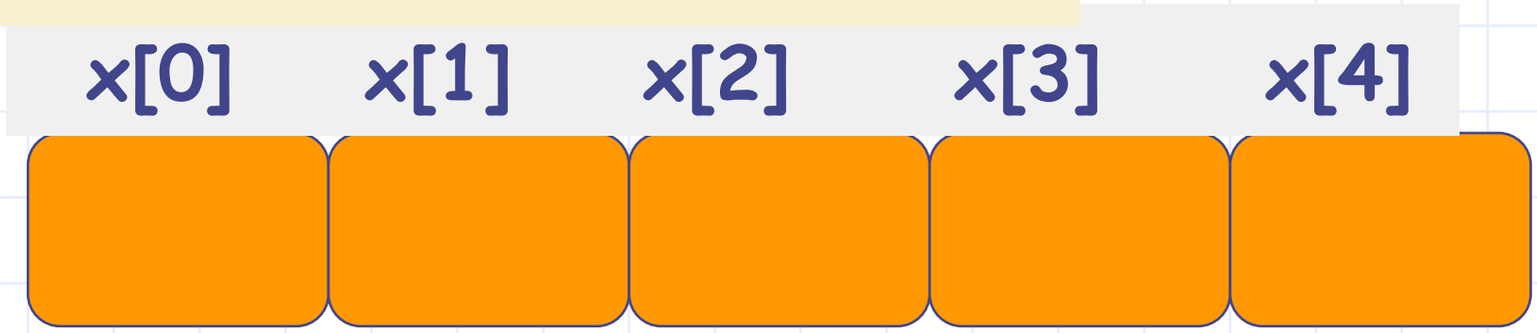


Consider program fragment:

```
int main() {
    int x[5];
        …
}
```

This defines an integer array named x of size 5.

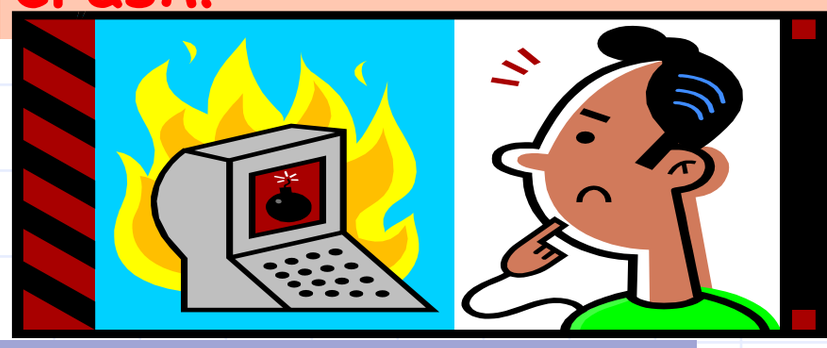
Five integer variables named x[0] x[1] … x[4] are allocated.



The variables x[0],x[1] … x[4] are integers, and can be assigned and operated upon like integers! OK, so far so good!

But what about x[5], x[6], … x[55]?
Can I assign to x[5], increment it, etc.?

NO! Program may crash!



Why? x[5], x[6], and so on are undefined. These are names but no storage has been allocated. Shouldn't access them!

Q: Shouldn't I or couldn't I access array elements outside of the array range declared?

Ans: You can but shouldn't. Program may crash.


```
int main() {
    int x[5];
    x[0] =0;
    x[1] =1;
    ...
    x[4] =4;

    x[5] = 5;
    x[6] = 6;
}
```

All good

Both these statements are not recommended.

Will it compile? Yes, it will compile. C compiler may give a warning.

But, upon execution, the program may give "segmentation fault: core dumped" error or it may also run correctly and without error.

Practice Problem

- Write a program to read in an array of 5 floating point numbers. Compute its mean and compute the difference of each element from its mean.

- Input: 3 1 5 2 9
- Output: -2.00 -3.00 1.00 -2.00 5.00

Solution for Practice problem

```
#include <stdio.h>

int main()
{
    float arr[5];
    float sum=0, mean;
    for(int i=0; i<5; i++)
    {
        scanf("%f",&arr[i]);
        sum=sum+arr[i];
    }

    mean=_________;
    for(int i=0; i<5; i++)
    {
        arr[i] =_______________
        printf("%.2f ",arr[i]);
    }
    return 0;
}
```

Solution for Practice problem

```
#include <stdio.h>

int main()
{
    float arr[5];
    float sum=0, mean;
    for(int i=0; i<5; i++)
    {
        scanf("%f",&arr[i]);
        sum=sum+arr[i];
    }

    mean=sum/5;
    for(int i=0; i<5; i++)
    {
        arr[i] =_____________
        printf("%.2f ",arr[i]);
    }
    return 0;
}
```

Solution for Practice problem

```
#include <stdio.h>

int main()
{
    float arr[5];
    float sum=0, mean;
    for(int i=0; i<5; i++)
    {
        scanf("%f",&arr[i]);
        sum=sum+arr[i];
    }

    mean=sum/5;
    for(int i=0; i<5; i++)
    {
        arr[i] = arr[i]-mean;
        printf("%.2f ",arr[i]);
    }
    return 0;
}
```

Array example: print backwards

Problem:

Define a character array of size 100 (upper limit) and read the input character by character and store in the array until either

- 100 characters are read or
- EOF (End Of File) is encountered

Now print the characters backwards from the array.

Example Input 1

```
Me or
Moo
```

Output 1

```
ooM
ro eM
```

Example Input 2

```
Eena Meena Dika
```

Output 2

```
akiD aneeM aneE
```

Read and print in reverse

1. We will design the program in a top down fashion, using the main() function.
2. There will be two parts to main: read_into_array and print_reverse.
3. read_into_array will read the input character-by-character up to 100 characters or until the end of input.
4. print_reverse will print the characters in reverse.

Overall design

```
int main() {
        char s[100]; /* to hold the input */
        /* read_into_array */
        /* print_reverse */
        return 0;
}
```

Let us design the program fragment read_into_array.

Keep the following variables:

1. int count to count the number of characters read so far.
2. int ch to read the next character using getchar().

Note that getchar() has prototype **int getchar()**
since getchar() returns all the 256 characters and the
integer EOF

```
int ch;
int count = 0;
read the next character into ch using getchar();
while (ch is not EOF AND count < 100) {
        s[count] = ch;
        count = count + 1;
        read the next character into ch using getchar();
}
```

An initial design (pseudo-code)

```
int ch;
int count = 0;
read the next character into ch using getchar();
while (ch is not EOF AND count < 100) {
        s[count] = ch;
        count = count + 1;
        read the next character into ch using getchar();
}
```

initial design pseudo-code

```
int ch;
int count = 0;
ch = getchar();
while ( ch != EOF && count < 100) {
        s[count] = ch;
        count = count + 1;
        ch = getchar();
}
```

Translating the read_into_array pseudo-code into code.

Overall design

```
int main() {
    char s[100];
 /* read_into_array */
 /* print_reverse */
    return 0;
}
```

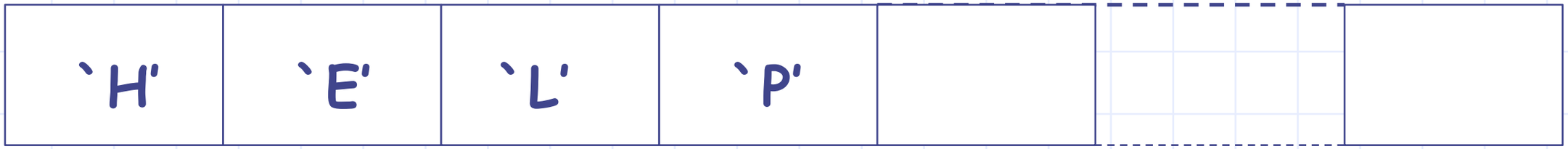
What is the value of count at the end of read_into_array?

Now let us design the code fragment **print_reverse**

Suppose input is

HELP<eof>

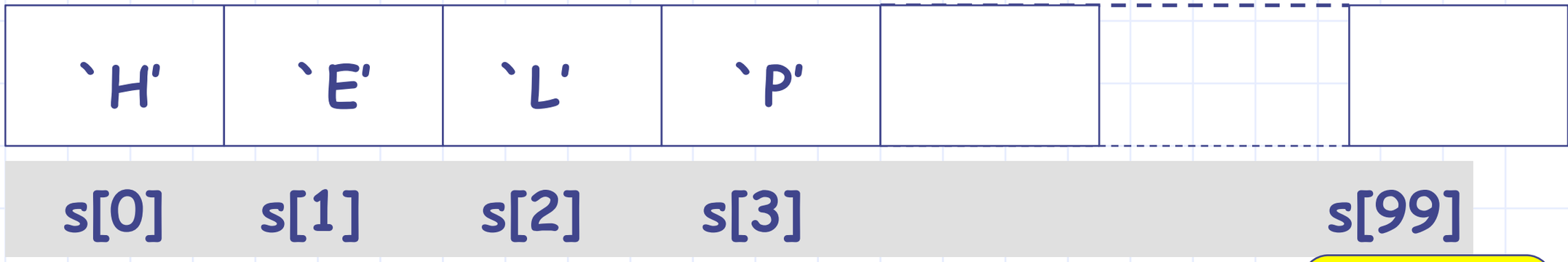
The array char s[100]




```
int i;
set i to the index of  last  character read.

while (i >= 0) {
    print s[i]
    i = i-1;       /* shift array index one to  left */
}
```

PSEUDO CODE

The array char s[100]




```
int i;
set i to index of the last character read.

while (i >= 0) {
    print s[i]
    i = i-1;
}
```

PSEUDO CODE

Translating pseudo code to C code: print_reverse

```
int i;
i = count-1;

while (i >=0) {
        putchar(s[i]);
        i=i-1;
}
```

Code for printing characters read in array in reverse

Putting it together

Overall design

```
int main() {
        char s[100];
        /* read_into_array */
        /* print_reverse */
        return 0;
}
```

The code fragments we have written so far.

```
int count = 0;
int ch;
ch = getchar();
while ( ch != EOF && count < 100) {
        s[count] = ch;
        count = count + 1;
        ch = getchar();
```

read_into_array code.

```
int i;
i = count-1;
while (i >=0) {
        putchar(s[i]);
        i=i-1;
}
```

print_reverse code

```
#include <stdio.h>
int main() {
      char s[100];
      int count = 0;
      int ch;
      int i;

      ch = getchar();
      while ( ch != EOF && count < 100) {
            s[count] = ch;
            count = count + 1;
            ch = getchar();
      }

      i = count-1;
      while (i >=0) {
            putchar(s[i]);
            i=i-1;
      }

      return 0;
}
```

```
/* the array of 100 char */
/* counts number of input chars read */
/* current character read  */
/* index for printing array backwards */
```

```
/*read_into_array */
```

```
/*print_in_reverse */
```

```c
#include <stdio.h>
int main() {
    char s[100];
    int count = 0;
    int ch;
    int i;
    while ( (ch=getchar()) != EOF &&
                    count < 100 )
    {       s[count] = ch;
            count = count + 1;
    }
    i = count-1;
    while (i >=0) {
            putchar(s[i]);
            i=i-1;
    }
    return 0;
}
```

/*read_into_array */

/*print_in_reverse */