

Counting number of students who scored marks-i

```
#include <stdio.h>

int main() {
    int numStudents, marks, i;
    int marksCount[26];

    scanf("%d", &numStudents);
    for (i = 0; i < numStudents; i++) {
        scanf("%d", &marks);
        marksCount[marks]++;
    }
    return 0;
}
```

Is the program correct?

Counting number of students who scored marks-i

```
#include <stdio.h>

int main() {
    int numStudents, marks, i;
    int marksCount[26];

    scanf("%d", &numStudents);
    for (i = 0; i < numStudents; i++) {
        scanf("%d", &marks);
        marksCount[marks]++;
    }
    return 0;
}
```

Is the program correct?

- Initialization of marksCount missing.
- What if the user enters marks outside the range?

Counting number of students who scored marks-i

```
#include <stdio.h>

int main() {
    int numStudents, marks, i;
    int marksCount[26] = {0};

    scanf("%d", &numStudents);
    for (i = 0; i < numStudents; i++) {
        scanf("%d", &marks);
        if (marks >= 0 && marks <= 25)
            marksCount[marks]++;
    }
    return 0;
}
```

Initializing an array

Different ways of initializing array.

- `int count[] = {10, 23, 50};`
Creates an array of 3 integers. `count[0]`, `count[1]`, `count[2]`.
- `int count[10] = {0};`

Initializing an array

Different ways of initializing array.

- `int count[] = {10, 23, 50};`
Creates an array of 3 integers. `count[0]`, `count[1]`, `count[2]`.
- `int count[10] = {0};`
- Using a loop to explicitly initialize the elements.

Initializing an array

Different ways of initializing array.

- `int count[] = {10, 23, 50};`
Creates an array of 3 integers. `count[0]`, `count[1]`, `count[2]`.
- `int count[10] = {0};`
- Using a loop to explicitly initialize the elements.

Common Mistake: Forgetting to initialize the elements of array.

Evaluating a polynomial

$$P(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

-
- n is the degree of a polynomial.
 - User provides $n + 1$ coefficients: $a_0, a_1, \dots, a_n$.
 - User provides the value of x at which polynomial has to be evaluated.
 - Evaluate the polynomial.
-

Evaluating a polynomial

$$P(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

- Evaluate each term separately.
 - n additions.
 - $n + (n - 1) + (n - 2) + \dots + 2 + 1 = \frac{n(n+1)}{2}$ multiplications.

Evaluating a polynomial

$$P(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

- Evaluate each term separately.
 - n additions.
 - $n + (n - 1) + (n - 2) + \dots + 2 + 1 = \frac{n(n+1)}{2}$ multiplications.
- $P(x) = a_0 + x(a_1 + x(a_2 + \dots + x(a_{n-1} + xa_n)))$
 - n additions.
 - n multiplications.

Method 1: Repeated multiplication

```
#include <stdio.h>

int main() {
    int x, n, i, j;
    int coeff[20];
    int value = 0;

    scanf("%d-%d", &n, &x);
    for (i = 0; i <= n; i++)
        scanf("%d", &coeff[i]);

    for (i = 0; i <= n; i++) {
        int term = coeff[i];
        for (j = 1; j <= i; j++)
            term = term * x;
        value = value + term;
    }

    printf("%d\n", value);
    return 0;
}
```

Method 2: Using `pow(x,i)`

Include `math.h`. Replace the repeated-multiplication loop by:

```
for (i = 0; i <= n; i++) {  
    value = value + coeff[i] * pow(x, i);  
}
```

Evaluating a polynomial

```
#include <stdio.h>

int main() {
    int x, n, i;
    int coeff[20];
    int value;

    scanf("%d-%d", &n, &x);
    for (i = 0; i <= n; i++) {
        scanf("%d", &coeff[i]);
    }

    /* Fill in your code here */

    printf("%d\n", value);
    return 0;
}
```

Horner's method

```
#include <stdio.h>

int main() {
    int x, n, i;
    int coeff[20];

    scanf("%d-%d", &n, &x);
    for (i = 0; i <= n; i++)
        scanf("%d", &coeff[i]);

    int value = coeff[n];
    for (i = n - 1; i >= 0; i--)
        value = value * x + coeff[i];

    printf("%d\n", value);
    return 0;
}
```

Character arrays

```
char name[20];
```

Different ways of initialization

- `char name[20] = "Avani";`
 - `char name[20] = {'A', 'V', 'A', 'N', 'I', 'null char'};`
-

```
char name[20];
```

Different ways of initialization

- `char name[20] = "Avani";`
- `char name[20] = {'A', 'V', 'A', 'N', 'I', '\0'};`
- `char name[20];`
 `scanf("%s", name);`

Character arrays

```
char name[20];
```

Different ways of initialization

- `char name[20] = "Avani";`
- `char name[20] = {'A', 'V', 'A', 'N', 'I', '\0'};`
- `char name[20];`
`scanf("%s", name);`
- `char name[20];`
`name = "AVANI";`

Incorrect!!
