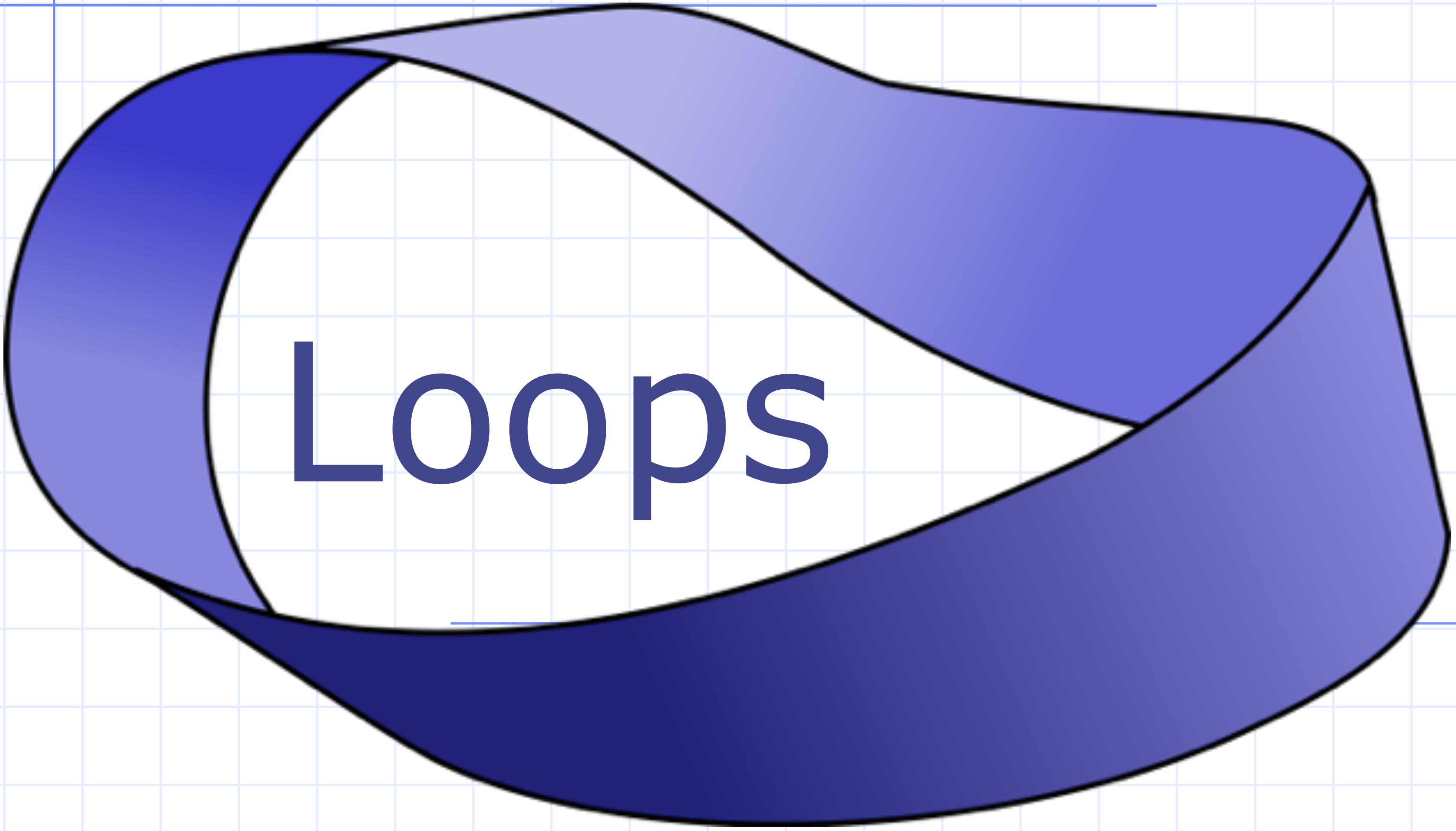
ESC101: Introduction to Computing



Loops

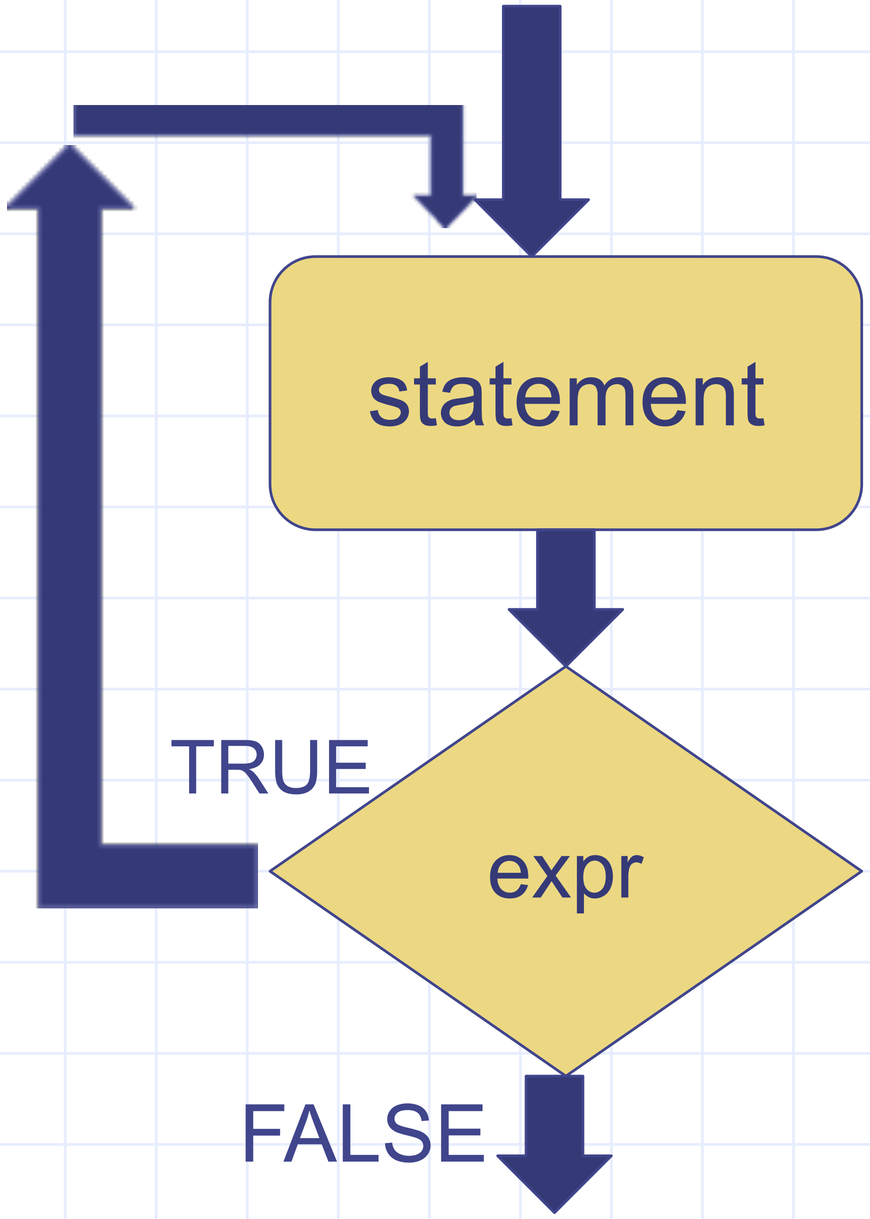
do-while loops

- do-while statement is a variant of while.

  General form:

```
do
    statement
while (expr);
```

- Execution:
  1. First execute statement.
  2. Then evaluate expr.
  3. If expr is TRUE then go to step 1.
  4. If expr is FALSE then break from loop
- Continuation of loop is tested after the statement.



Comparing while and do-while

- In a while loop the body of the loop may not get executed even once, whereas, in a do-while loop the body of the loop gets executed at least once.
- In the do-while loop structure, there is a semicolon after the condition of the loop.
- Rest is similar to a while loop.

Comparative Example

- Problem: Read integers and output each integer until -1 is seen (include -1 in output).
- The program fragments using while and do-while.

Using while

```
int a;/*current int*/

scanf("%d",&a);
while (a != -1) {
   printf("%d\n", a);
   scanf("%d", &a);
}
printf("%d\n", a);
```

Using do-while

```
int a; /*current int*/

do {
   scanf("%d", &a);
   printf("%d\n", a);
} while (a != -1);
```

Comparative Example

- The while construct and do-while are equally expressive
  - whatever one does, the other can too.
  - but one may be *more readable* than other.

Using do-while

```
int a; /*current int*/

do {
   scanf(“%d”, &a);
   printf(“%d\n”, a);
} while (a != -1);
```

Using while

```
int a;/*current int*/

scanf(“%d”,&a);
while (a != -1) {
   printf(“%d\n”, a);
   scanf(“%d”, &a);
}
printf(“%d\n”, a);
```

Practice Problem

- Write a program to use do-while to print the squares of the first n integers

- Given number 5 - output is
    - 1
    - 4
    - 9
    - 16
    - 25

Write a program that prints squares of first n integers

```
#include <stdio.h>
int main()
{
   int n,i=___;
   scanf("%d",&n); //assuming n>0
   do{
      printf("%d\n",i*i);
      i = i+1;
   } while(____);
   return 0;
}
```

Write a program that prints squares of first n integers

```
#include <stdio.h>
int main()
{
   int n,i=1;
   scanf("%d",&n); //assuming n>0
   do{
      printf("%d\n",i*i);
      i = i+1;
   } while(____);
   return 0;
}
```

Write a program that prints squares of first n integers

```
#include <stdio.h>
int main()
{
    int n,i=1;
    scanf("%d",&n); //assuming n>0
    do{
        printf("%d\n",i*i);
        i = i+1;
    } while(i<=n);
    return 0;
}
```

Practice Problem

- Add numbers till -1 is not seen. Use do while

Add numbers until -1 using do while

```
int a;
int s;
s = 0; // not seen any a yet
do {
   scanf(“%d”, &a);      // read into a
     s = s + a;
} while (a !=  -1)
 // one could print s here etc.
```

Add numbers until -1 using do while

```
int a=0;
int s;
s = 0; // not seen any a yet
do {
    s = s + a;
    scanf("%d", &a);      // read into a
} while (a !=  -1)
 // one could print s here etc.
```

For Loop

For Loop

Print the sum of the reciprocals of the first 100 natural numbers.

```
int i;                        // counter from 1..100
float rsum = 0.0; // the sum

// the for loop
for ( i=1; i<=100; i=i+1 ) {
   rsum = rsum +  (1.0/i);
}
printf(“sum is %f “, rsum);
```

For loop in C

- General form

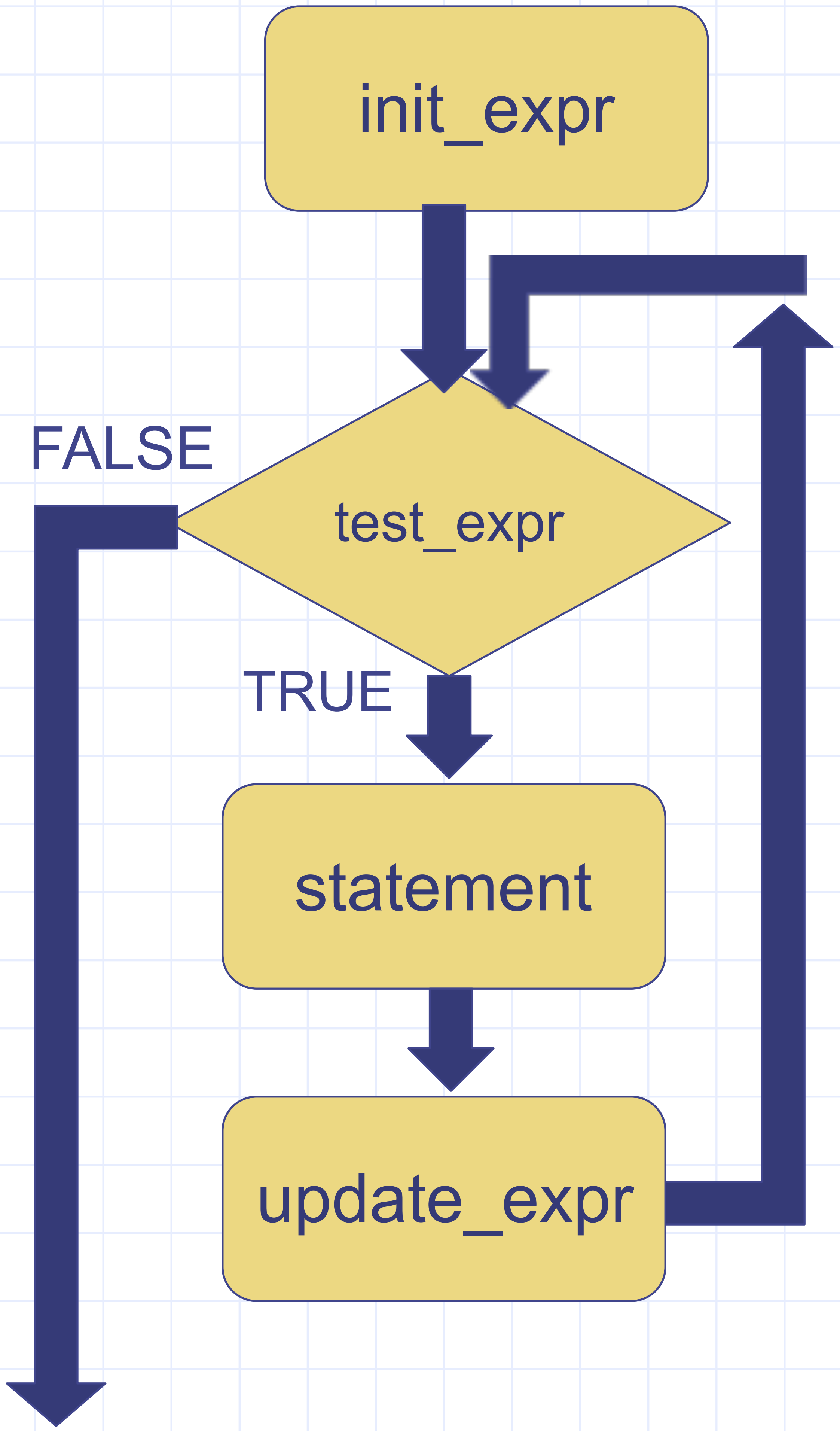
```
for (init_expr; test_expr;
update_expr)
    statement;
```

- init_expr is the initialization expression.
- update_expr is the update expression.
- test_expr is the expression that evaluates to either TRUE (non-zero) or FALSE (zero).
- statement is the work to repeat (can be multiple statements in {…})

For loop in C

```
for (init_expr; test_expr; update_expr)
     statement;
```

1. **First evaluate init_expr;**
2. **Evaluate test_expr;**
3. **If test_expr is TRUE then**
   a) **execute statement;**
   b) **execute update_expr;**
   c) **go to Step 2.**
4. **if test_expr is FALSE then break from the loop**



```
int i;
float rsum = 0.0;

for (i=1; i<=4; i=i+1) {
   rsum = rsum + (1.0/i);
}

printf("sum is %f", rsum);
```

1. Evaluate init_expr; i.e., i=1;
2. Evaluate test_expr i.e., i<=4 TRUE
3. Enter body of loop and execute.
4. Execute update_expr; i=i+1; i is 2
5. Evaluate test_expr i<=4: TRUE
6. Enter body of loop and execute.
7. Execute i=i+1; i is 3
8. Evaluate test_expr i<=4: TRUE
9. Enter body of loop and execute.
10. Execute i=i+1; i is 4
11. Evaluate test_expr i<=4 TRUE
12. Enter body of loop and execute.
13. Execute i=i+1; i is 5
14. Evaluate test_expr i<=4 FALSE
15. Exit loop & jump to printf

```
sum is 2.083333
```

For loop in terms of while loop

```
for (init_expr; test_expr; update_expr)
    statement;
```

- Execution is (almost) equivalent to

```
init_expr;
while (test_expr) {
    statement;
    update_expr;
}
```

- Almost? Exception if there is a continue; inside statement– this will be covered later.
- Both are equivalent in power.
- Which loop structure to use, depends on the convenience of the programmer.

Example: Geometric Progression

- Given positive real numbers r and a, and a positive integer, n, the n^{th} term of the geometric progression with a as the first term and r as the common ratio is ar^{n-1}.
- Write a program that given r, a, and n, displays the first n terms of the corresponding geometric progression.

```
#include<stdio.h>
int main(){
    int n, i;     float r, a, term;

    // Reading inputs from the user
    scanf("%f", &r);
    scanf("%f", &a);
    scanf("%d", &n);
    term = a;
    for (i=1; i<=n; i=i+1) {
        printf("%f\n", term); // Displaying ith term
        term = term * r;      // Computing (i+1)th term
    }
    return 0;
}
```

```
#include<stdio.h>
int main(){
    int n, i;     float r, a, term;

    // Reading inputs from the u
    scanf("%f", &r);
    scanf("%f", &a);
    scanf("%d", &n);
    term = a;
    for (i=1; i<=n; i=i+1) {
        printf("%f\n", term); // Displaying ith term
        term = term * r;        // Computing (i+1)th term
    }
    return 0;
}
```

Careful: Changing the order of statements changes the meaning of the program.
Computation of

$a, ar, \dots, ar^{n-1}$ vs.

$ar, ar^2, \dots, ar^n$

Practice Problem

- Write a program to use for loop to print the squares of the first n integers

- Given number 5 - output is
    - 1
    - 4
    - 9
    - 16
    - 25

Write a program that prints the square of the first n numbers using for loop

```c
#include <stdio.h>

int main()
{
        int i, n;
        scanf("%d",&n);
        for( i=0; i<n; i++)
                printf("%d ",i*i);
        return 0;
}
```

Input: 5

Output: 0 1 4 9 16

Write a program that prints the square of the first n numbers using for loop

```
#include <stdio.h>

int main()
{
        int i, n;
        scanf("%d",&n);
        for( i=1; i<=n; i++)
                printf("%d ",i*i);
        return 0;
}
```

Input: 5

Output: 1 4 9 16 25

Practice Problem

- Write a program to count the number of zeros in a given input integer

- Input: 10100
- Output: There are 3 zeros in the number

Write a program that counts and prints the number of zeros in an input integer

```
#include <stdio.h>
int main()
{
   int n,cnt=0;
   scanf("%d",&n);
   for(___; ____ ;____)
   {
      if(n%10 == 0)
         cnt=cnt+1;
   }
   printf("There are %d zeros in the number\n",cnt);
   return 0;
}
```

Write a program that counts and prints the number of zeros in an input integer

```
#include <stdio.h>
int main()
{
   int n,cnt=0;
   scanf("%d",&n);
   for(   ; ____ ;____)
   {
      if(n%10 == 0)
         cnt=cnt+1;
   }
   printf("There are %d zeros in the number\n",cnt);
   return 0;
}
```

Write a program that counts and prints the number of zeros in an input integer

```
#include <stdio.h>
int main()
{
   int n,cnt=0;
   scanf("%d",&n);
   for(   ; n>0 ;____)
   {
      if(n%10 == 0)
         cnt=cnt+1;
   }
   printf("There are %d zeros in the number\n",cnt);
   return 0;
}
```

Write a program that counts and prints the number of zeros in an input integer

```
#include <stdio.h>
int main()
{
   int n,cnt=0;
   scanf("%d",&n);
   for(   ; n>0 ; n=n/10)
   {
      if(n%10 == 0)
         cnt=cnt+1;
   }
   printf("There are %d zeros in the number\n",cnt);
   return 0;
}
```

Nested Loops

- Loop with in a loop
- Many iterations of inner loop ⇒ One iteration of outer loop