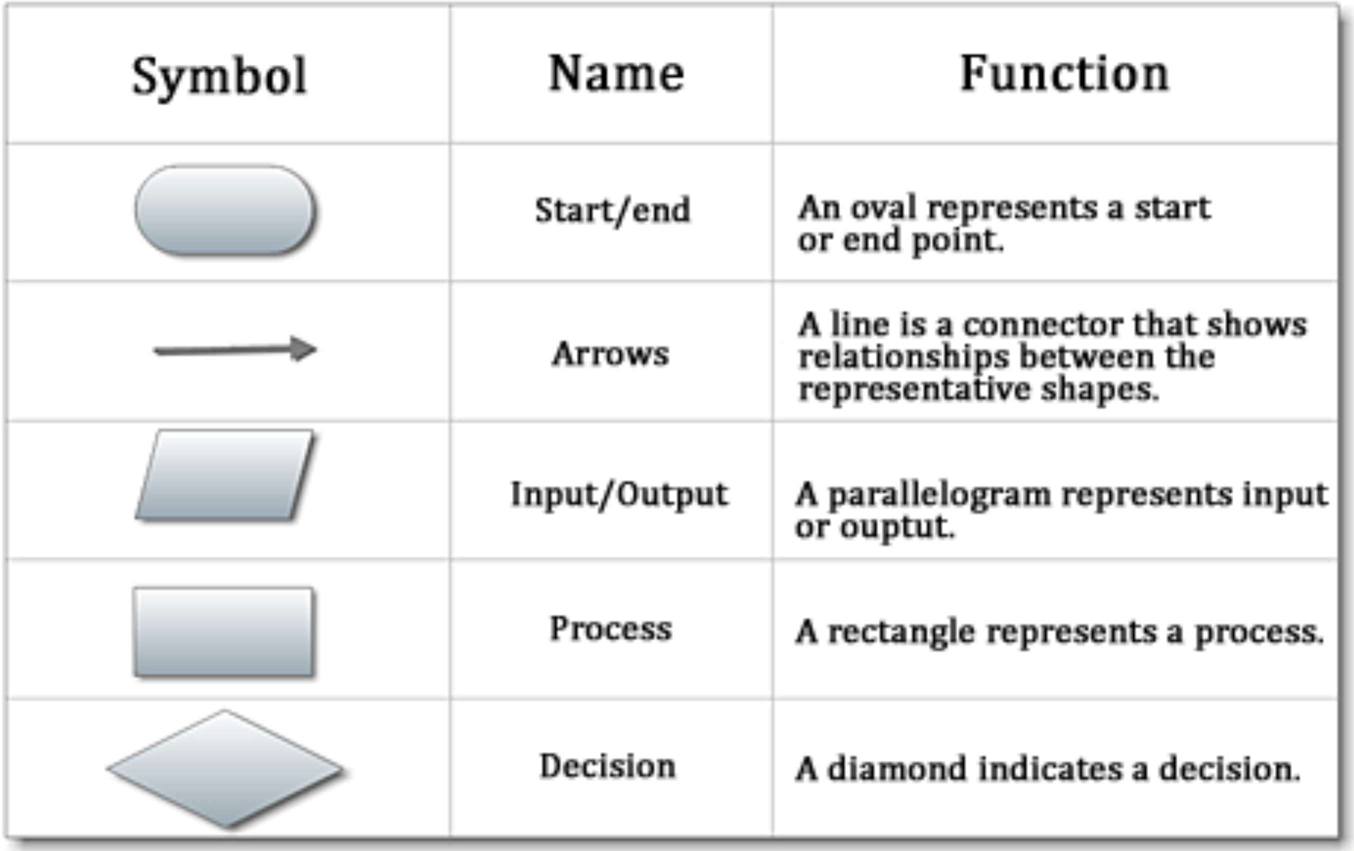
Program Design

Lecture 9

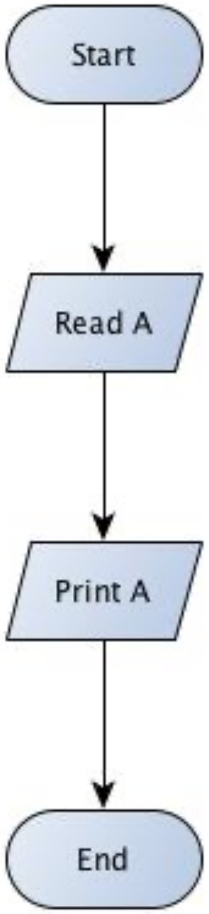
Flowcharts



Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point.
→	Arrows	A line is a connector that shows relationships between the representative shapes.
▱	Input/Output	A parallelogram represents input or ouptut.
▭	Process	A rectangle represents a process.
◇	Decision	A diamond indicates a decision.

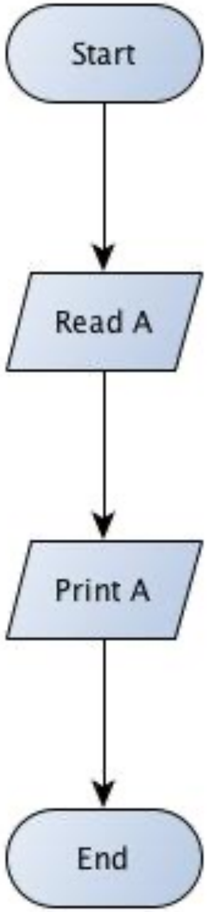
C version

```
#include <stdio.h>
int main()
{
        int A;
        printf(" please enter a value for A”);
        scanf("%d", &A);
        printf("You entered %d", A);
        return 0;
}
```



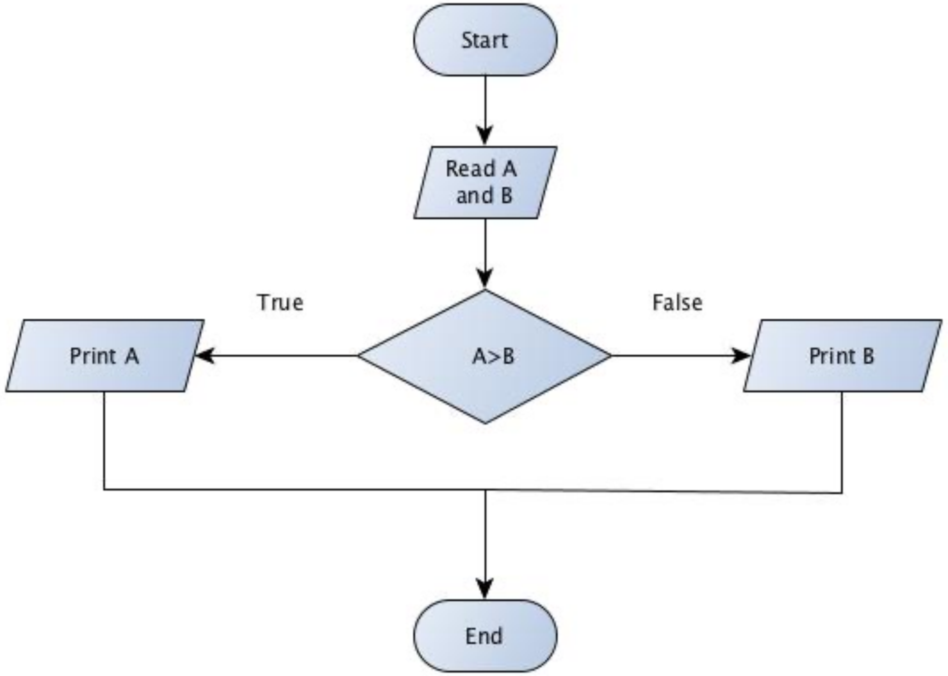
Python version

```
A = input(‘please enter a value for A’)
print(A)
```



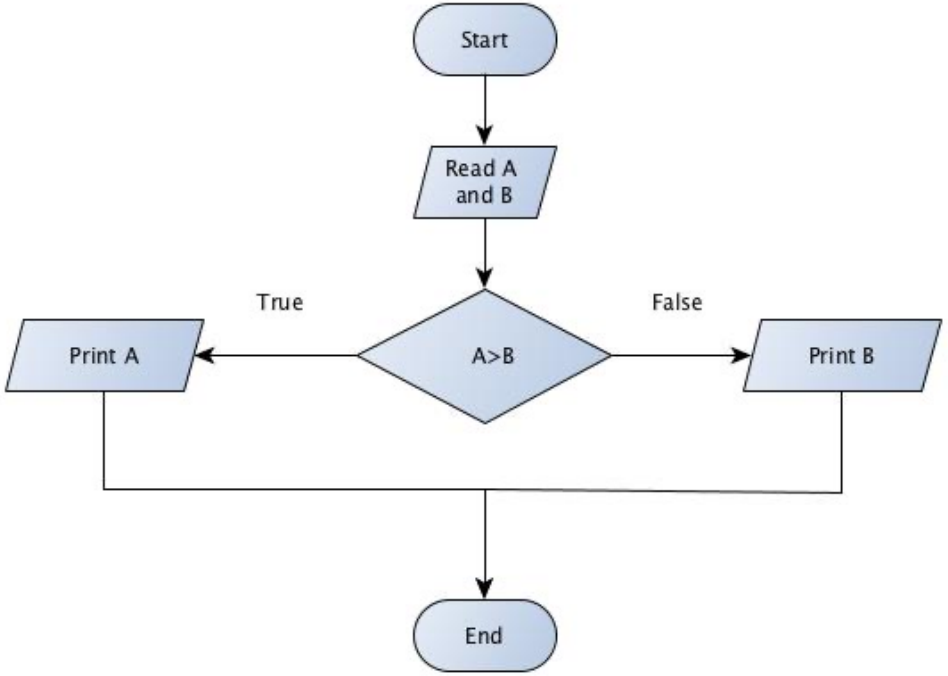
C version

```c
#include <stdio.h>
int main()
{
        int A;
        int B;
        printf("Please enter number A:");
        scanf("%d", &A);
        printf("Please enter number B:");
        scanf("%d", &B);
        if(A > B)
        {
                printf(“The biggest is %d", A);
        }
        else {
                printf(“The biggest is %d",B);
        }
return 0;
}
```



Python version

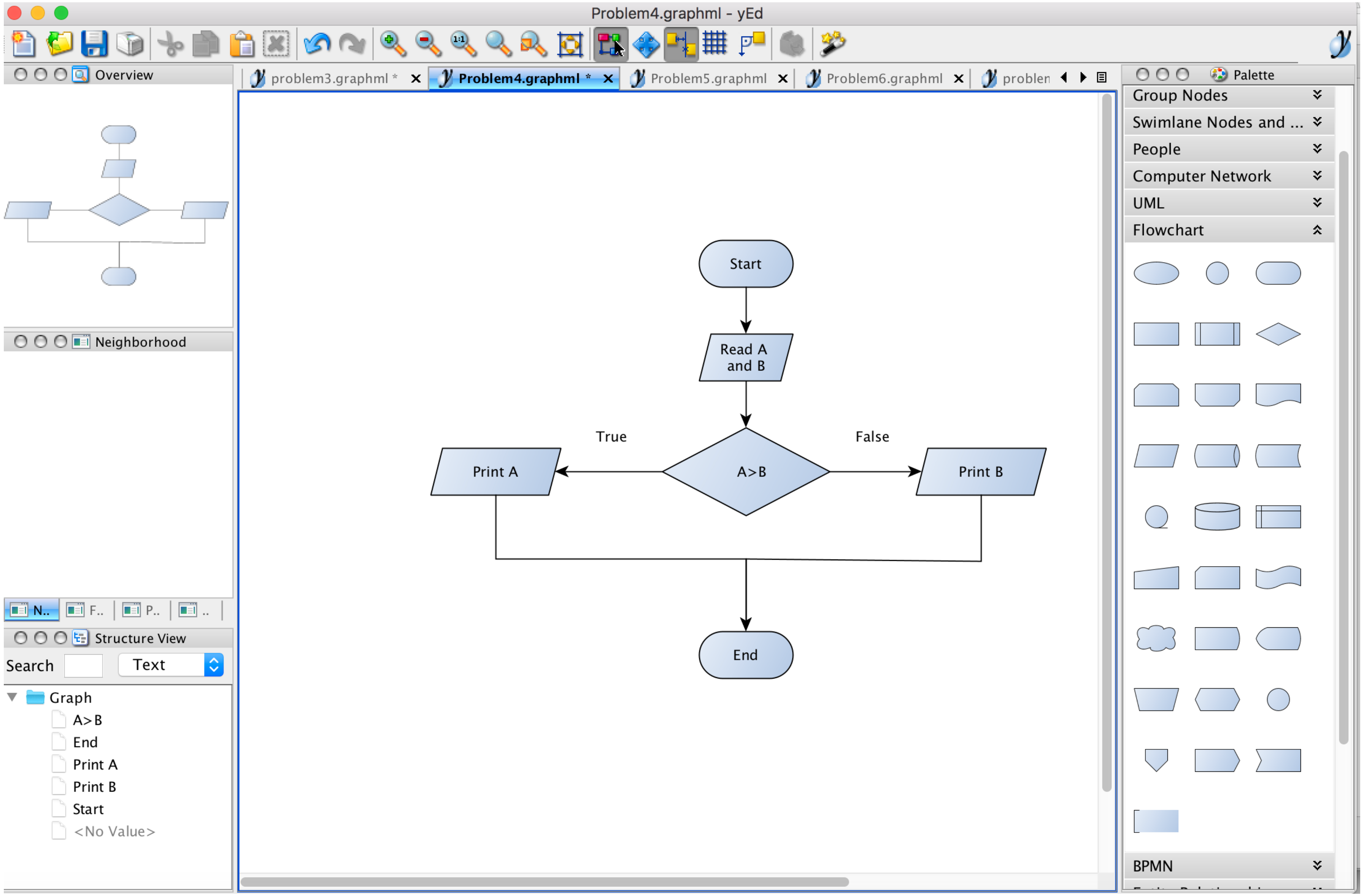
```
A = input(‘please enter a value for A’)
B = input(‘please enter a value for B’)
if A>B:
   print(A)
else
   print(B)
```



Programs for Flowcharts

- yEd
- Dia
- Microsoft Visio
- Draw.io
- Word

yEd



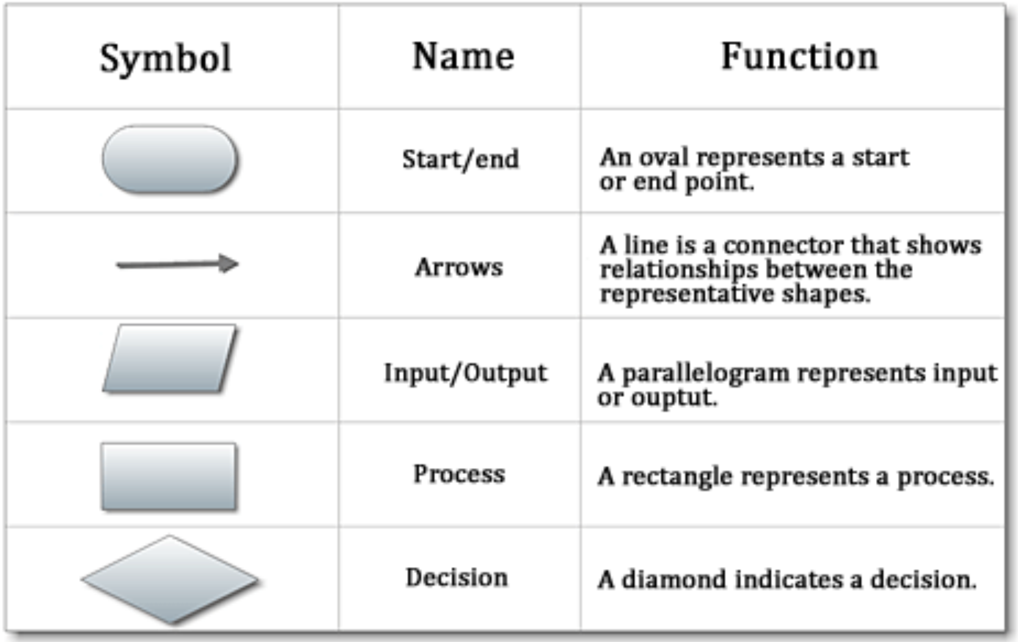
https://www.yworks.com/products/yed

More examples on flowcharts

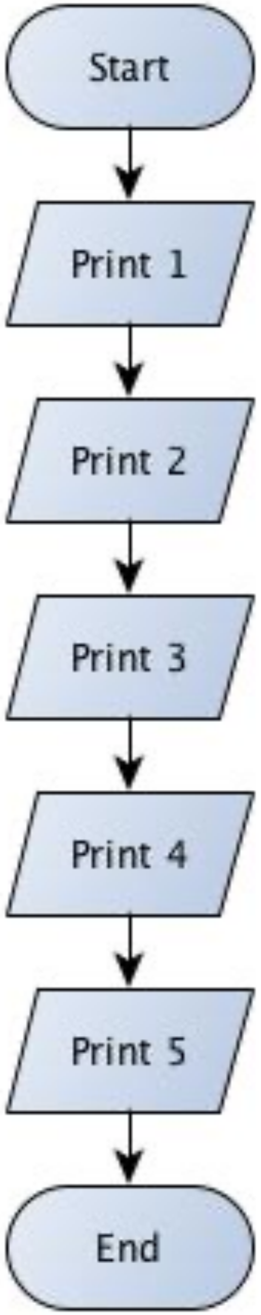
Flowcharts

So let’s say we want to express the following algorithm:

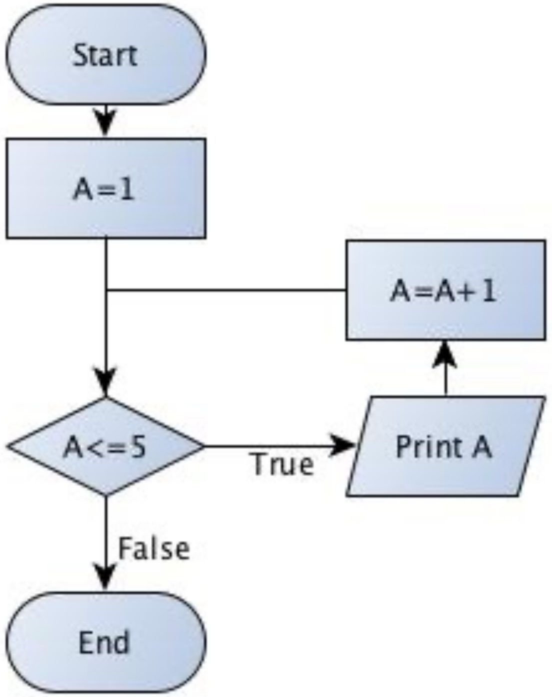
Print out the numbers from 1 to 5

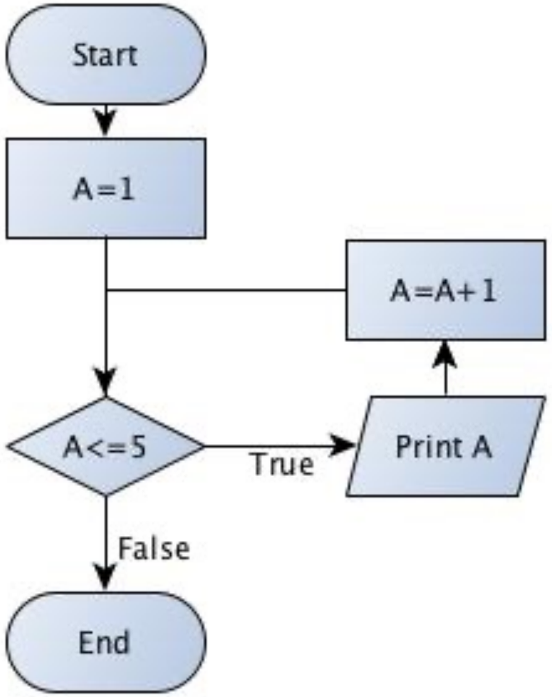


Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point.
→	Arrows	A line is a connector that shows relationships between the representative shapes.
▱	Input/Output	A parallelogram represents input or ouptut.
▭	Process	A rectangle represents a process.
◇	Decision	A diamond indicates a decision.


Start
Print 1
Print 2
Print 3
Print 4
Print 5
End

- Or alternatively…

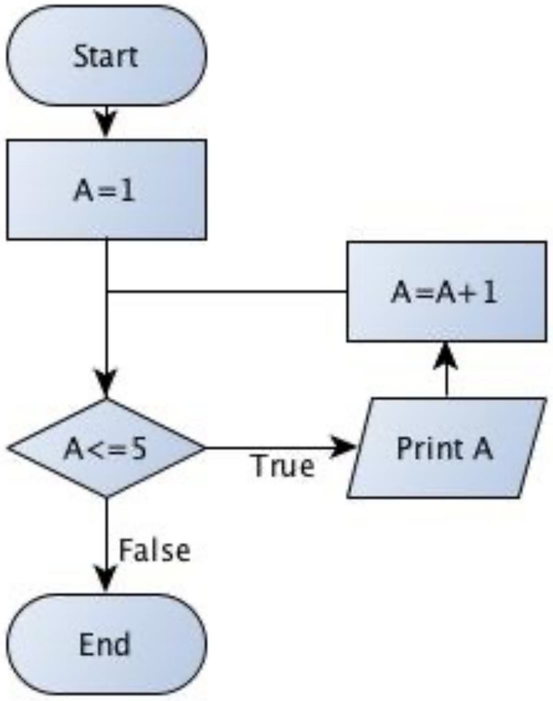




Before you put any other code in a loop you should write just this to make sure your loop is iterating as many times as you think it should – not one too many or one less – that's a common loop bug

Trace

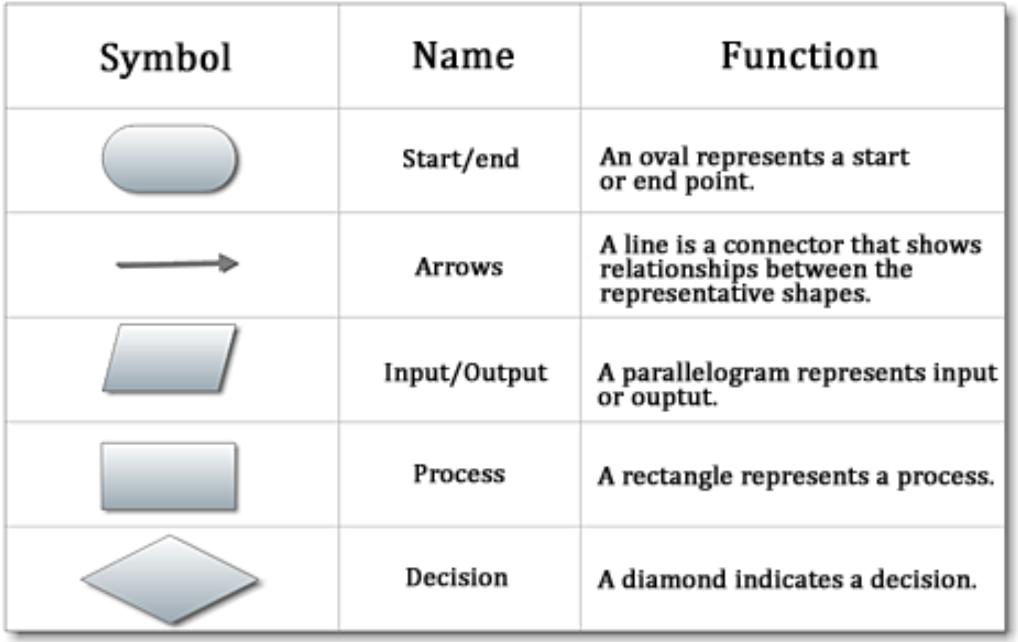
A=1	A=1	Is A<=5	Yes
A=A+1	A=2	Is A<=5	Yes
A=A+1	A=3	Is A<=5	Yes
A=A+1	A=4	Is A<=5	Yes
A=A+1	A=5	Is A<=5	Yes
A=A+1	A=6	Is A<=5	No



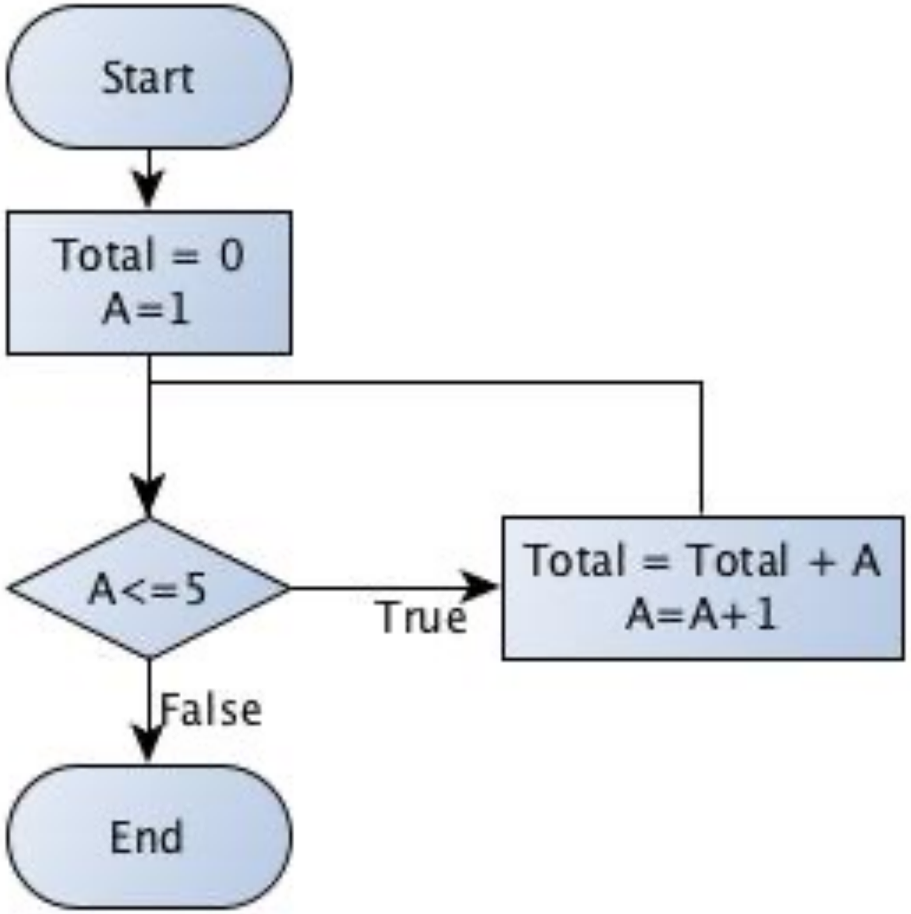
Flowcharts

So let’s say we want to express the following algorithm:

Add up the numbers from 1 to 5



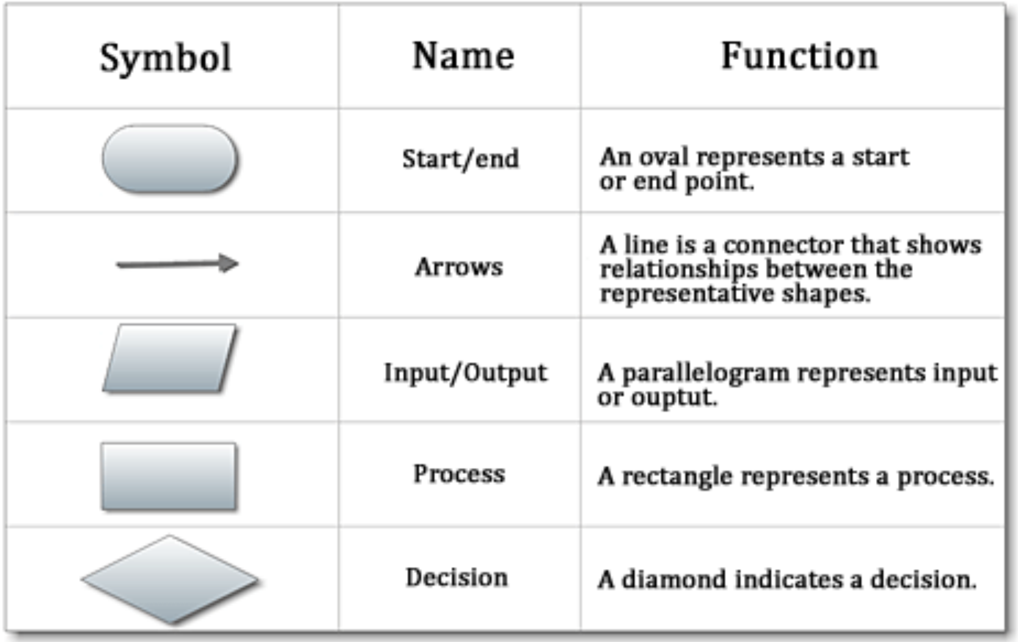
Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point.
→	Arrows	A line is a connector that shows relationships between the representative shapes.
▱	Input/Output	A parallelogram represents input or ouptut.
▭	Process	A rectangle represents a process.
◇	Decision	A diamond indicates a decision.


Start
Total = 0
A=1
A<=5
True
Total = Total + A
A=A+1
False
End

Flowcharts

So let’s say we want to express the following algorithm:

Read in a number and check if it’s a prime number.



Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point.
→	Arrows	A line is a connector that shows relationships between the representative shapes.
▱	Input/Output	A parallelogram represents input or ouptut.
▭	Process	A rectangle represents a process.
◇	Decision	A diamond indicates a decision.

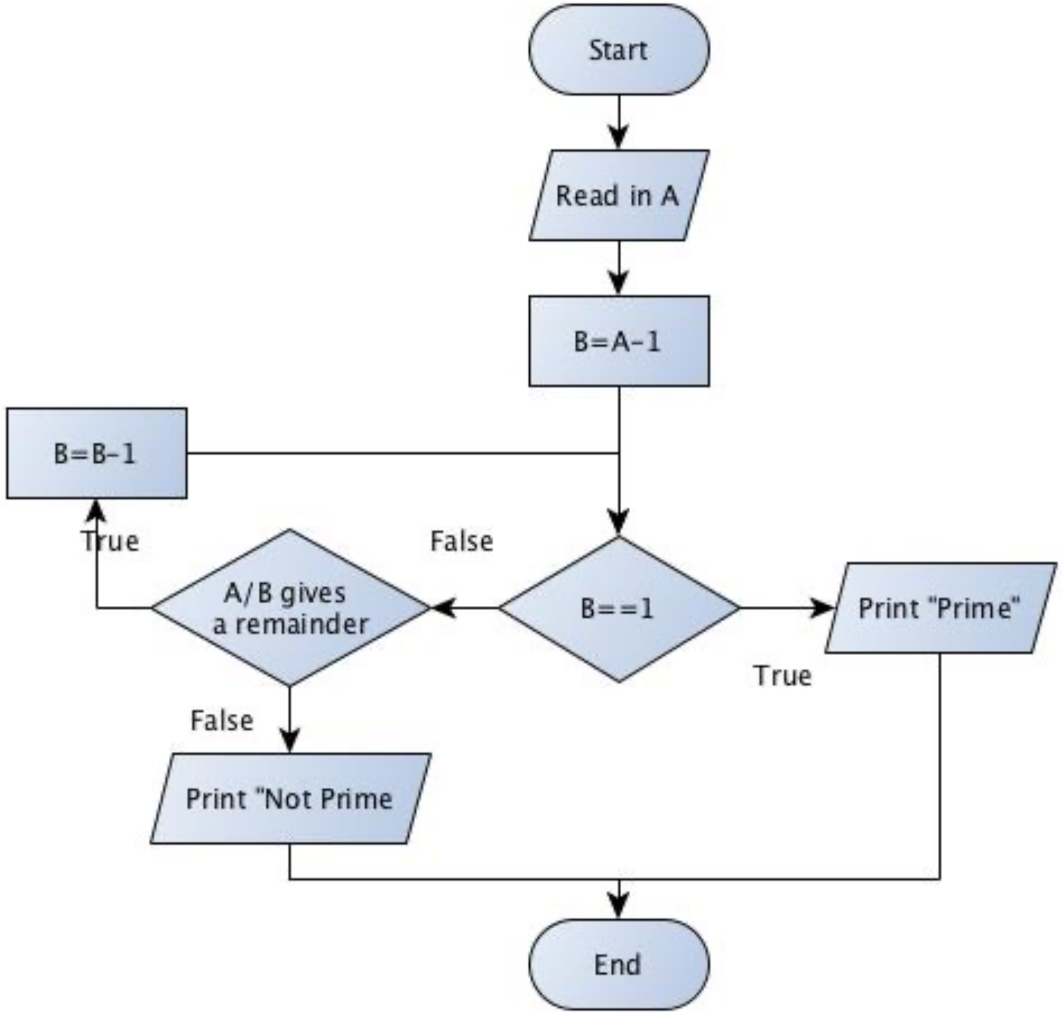
- A prime number is a number that’s only divisible by itself and 1, e.g. 1, 2, 3, 5, 7...
- Or to put it another way, every number other than itself and 1 gives a remainder, e.g. For 7, if 6, 5, 4, 3, and 2 give a remainder then 7 is prime.

What is a prime number?

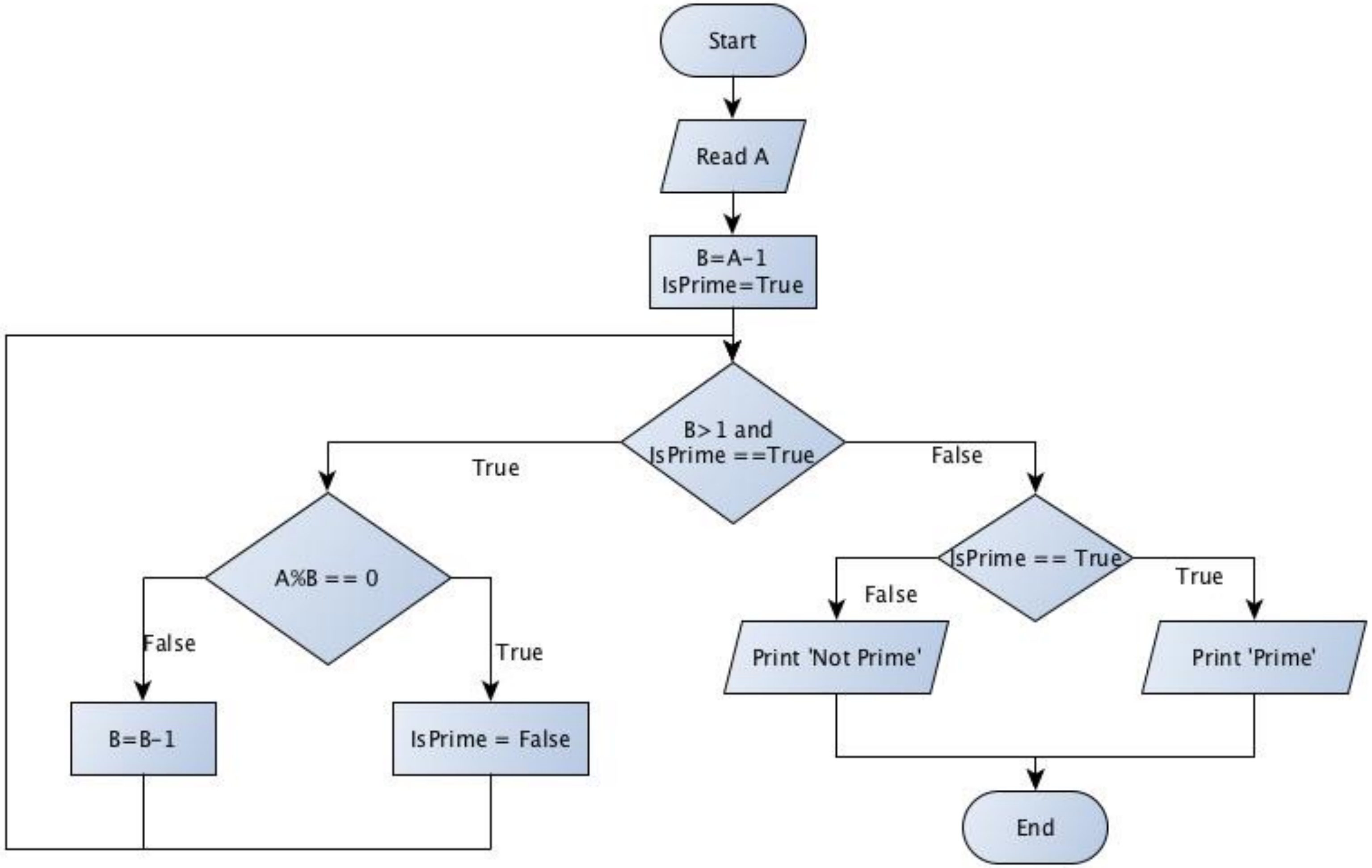
So all we need to do is divide 7 by all numbers less than it but greater than 1, and if all of them have a remainder, we know it is a prime.

So,

- If the number is 7, as long as 6, 5, 4, 3, and 2 give a remainder, 7 is a prime.
- If the number is 9, we know that 8, 7, 6, 5, and 4, all give remainders, but 3 does not give a remainder, it goes evenly into 9 so we can say 9 is not prime.
- in general - if the number is A, as long as A-1, A-2, A-3, A- 4, ... 2 give a remainder, A is a prime.


Start
Read in A
B=A-1
B=B-1
True
False
A/B gives a remainder
B==1
Print "Prime"
True
False
Print "Not Prime
End

Structured version

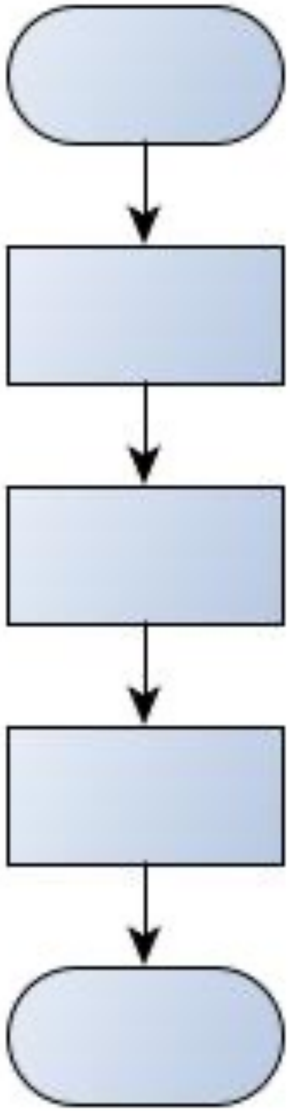


Our basic structures

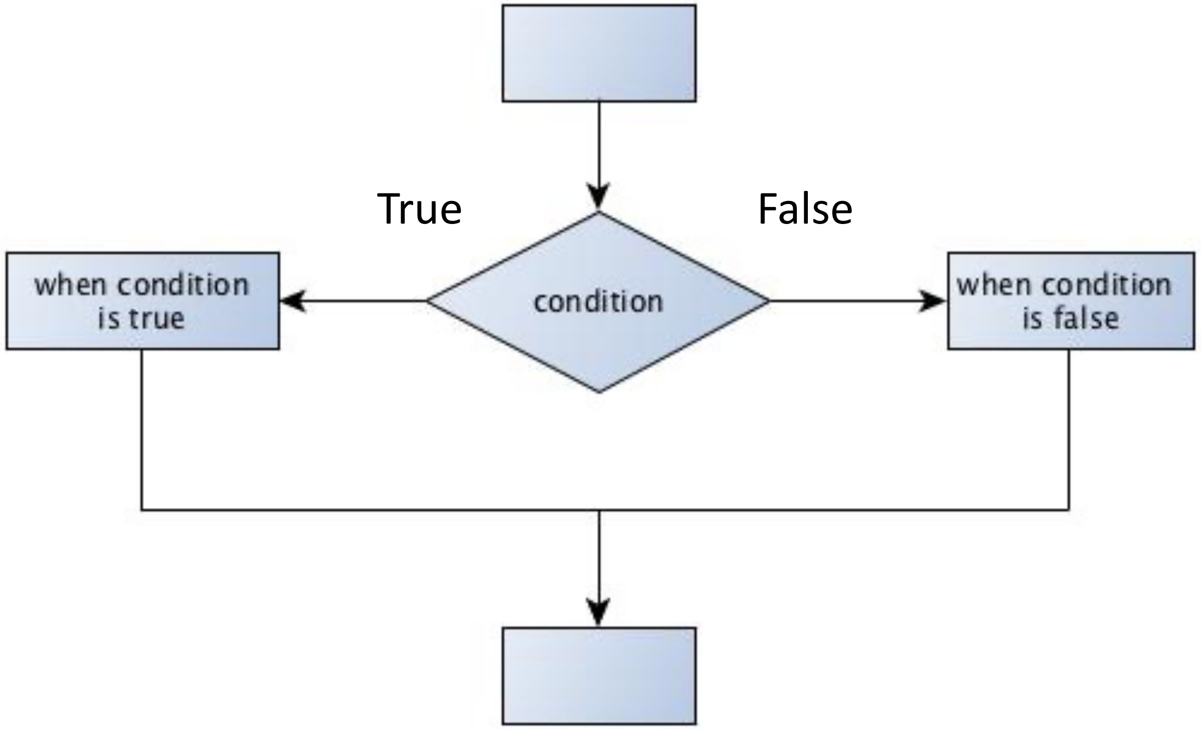
- Sequential statements
    - Input and Output
    - Variable creation and assignment
    - Arithmetic operations

- Decisions
    - IF statements
- Iterations
    - WHILE and FOR loops

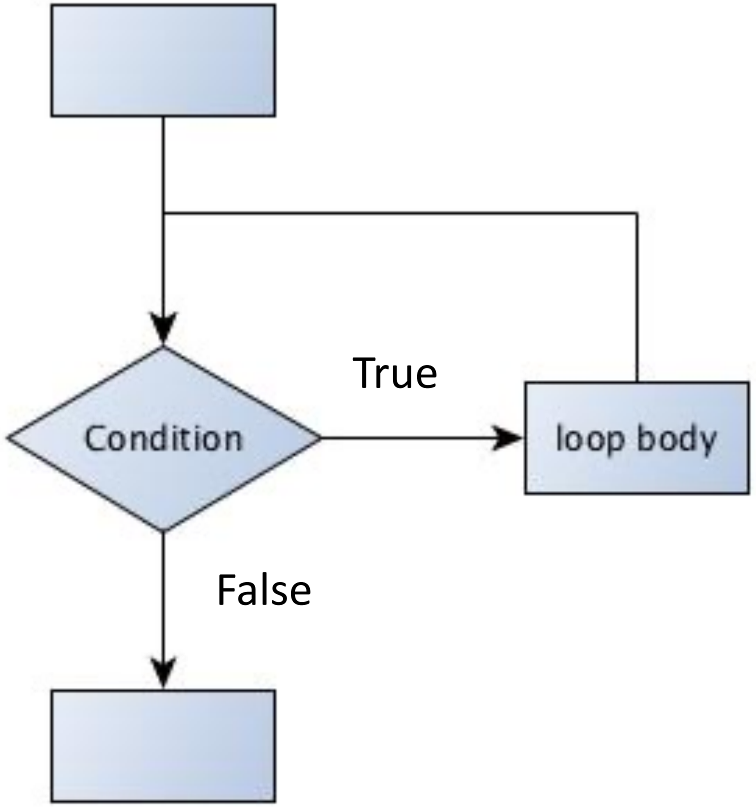
Sequence



If


True
False
when condition is true
condition
when condition is false

Loop


True
Condition
loop body
False