Why learn programming?

- It is really technical common sense!
- Software Engineers get great pay!
- Less stressful compared to several other high paying jobs - ok to do mistakes & learn from them
- Computer touches our lives more & more every day – it is good to know programming, even if you are in other fields.
- More component based programming → always room for simple programs to do big tasks!
- Software design focuses more on integration now, than writing everything from scratch.

Analogy: Learning to ride bicycle

- Difficulties for beginners:
  - Learning to balance & go forward together
- Difficulties for experienced folks:
  - Nothing specific.

Solution for beginners

- Training wheels
- Helmet

- Makes learning enjoyable and safe!

- Similar difficulties are there while learning to program in a computer.

Learning to program: Difficulties for beginners

1. Syntax errors

- struggle for hours to fix syntax errors
- Loose confidence
- Frustrating experience
- Run away & never come back if possible!

2. Logic errors

Not a serious issue.

Difficulties for experienced programmers

Logic errors

Continuous learning

Solution

- Visual Programming Tools to teach programming concepts without encountering syntax errors.
- Focus on the logic first & build confidence.

Free Visual Programming Tools

Tool	Provider	Web-site
Alice 2.3	Carnegie Mellon University	www.alice.org
Scratch	MIT	scratch.mit.edu
Snap!	UCBerkeley	byob.berkeley.edu
Lego MindStorm	Lego	mindstorms.lego.com
Several more…		

A few bit advanced tools

Tool	Provider	Web-site
Alice 3.1	Carnegie Mellon University	www.alice.org
JavaScript	Khan Academy	www.khanacademy.org/cs
Racket	UCBerkeley	wescheme.org
Greenfoot	Ukent	www.greenfoot.org
Several more…		

Programming Concepts based on every day activities

A few examples

- Recipe to make favorite food
- Assembly instructions for a toy
- Getting ready in the morning to go to school

What is common about these activities?

Sequence

Programming concepts: Sequence structure

instruction 1;

instruction 2;

instruction 3;

…

A few more examples

- Go to movie or study?
- Eat salad or sandwich?
- Go to job or go for higher studies?

What is the common thing here?

Selection or IF statement

Selection structure

```
IF condition is true THEN
    do this;
ELSE
    do that;
ENDIF
```

A few more examples

- Eat chips from a packet
- Go on a shopping spree with lot of cash!
- Take an exam that has several questions

What is the common thing here?

Repetition / Loops

Repetition structure

```
WHILE (more items to process)
        process the next item;
ENDWHILE

FOR  month = 1 to 12
      do monthly processing
ENDFOR
```

Programming Concepts

- Structures: Sequence, Selection & Repetition
- Foundation for Programming
- Every complex program is only a combination of these structures.

More things we do…

- Use a box to move lots of things from one room to another
- Carry a pack of candies to class on your birthday!

- What is the common thing here?

Collection / Arrays

Arrays

- enable us to store data of similar type together.
- enables us to handle varying size data.
- Lines of code do not increase with more data!

```
FOR each item in array
        add item to total
ENDFOR
```

Even more things we do…

- Get phone call when you are driving a car
- Friend knocks on the door when you are watching a movie

What is the common thing here?

Interrupts / events

Event driven programming

- Suspend current processing to process the event, or process it in parallel.
- “Regular” processing flow & separate processing routine for each event.

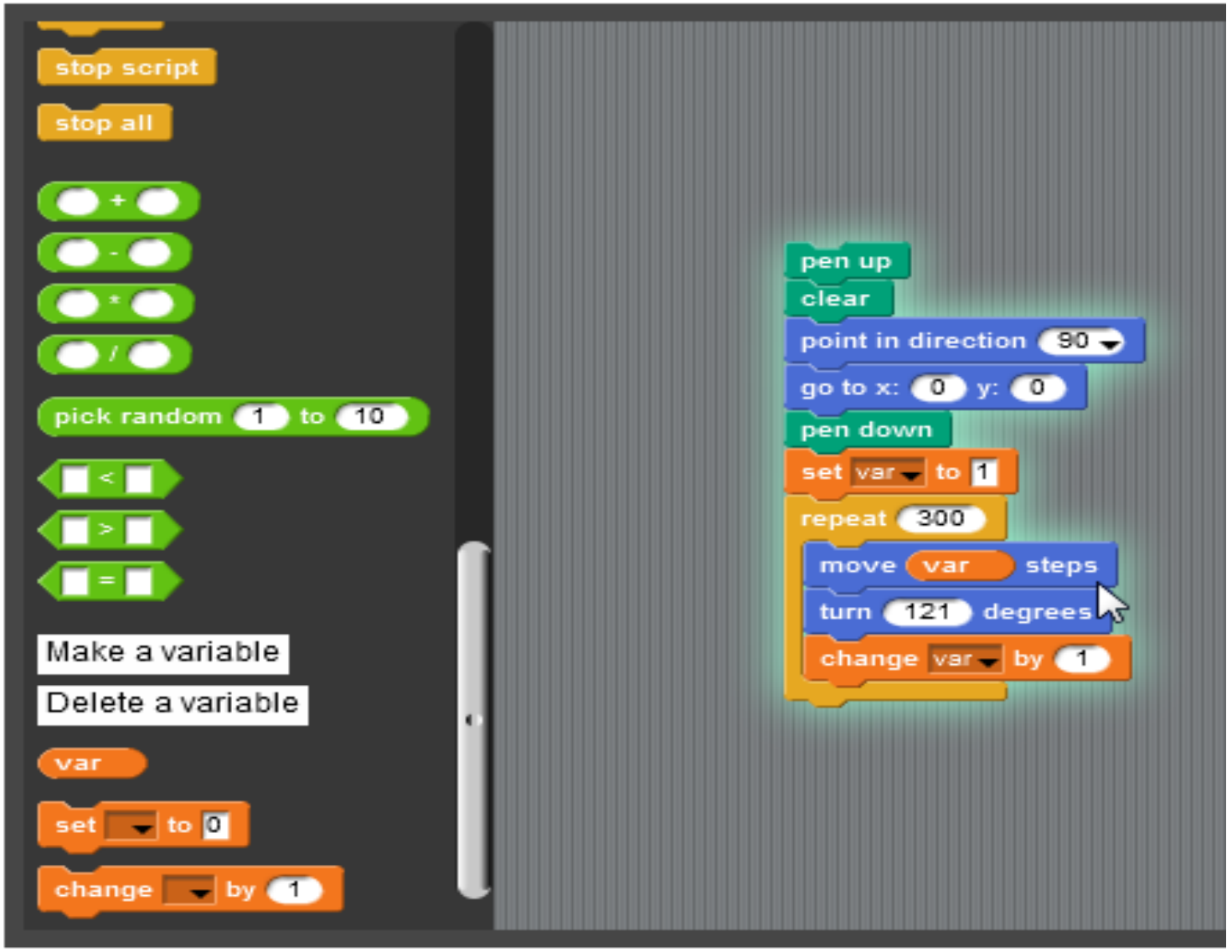
Object Oriented Programming (OOP)

- Models the real-world better
- Concepts learned from the manufacturing industry

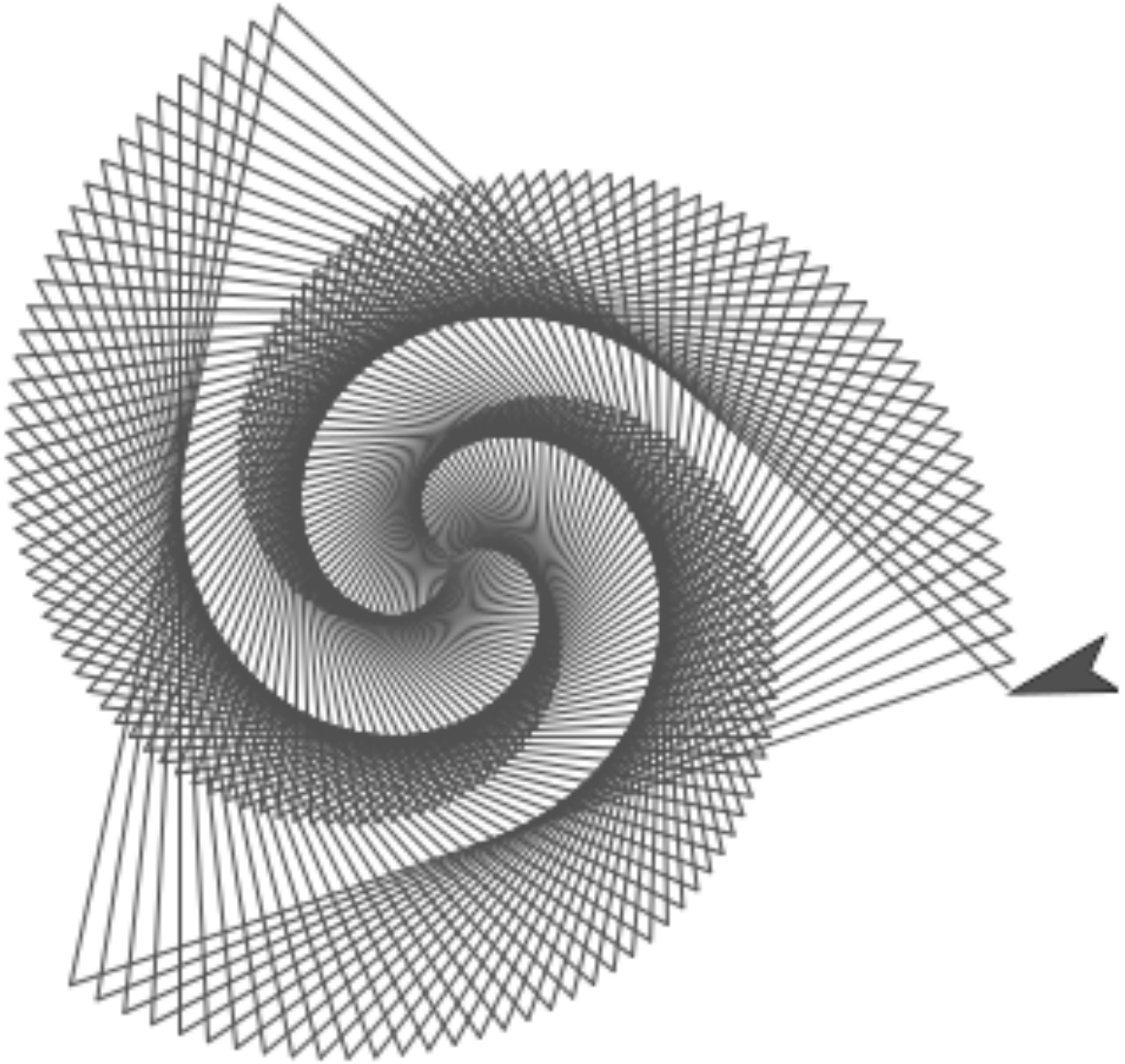
Alice demo

www.alice.org

Sample program in Snap 4.0



Program Output



10 ways to make learning to program enjoyable & efficient!

1. Take time to learn!

- Each person may learn at different pace.
- Each person has a different learning style.
- Learning C/C++/Java directly is NOT recommended. What is the hurry?
- If everything looks cryptic, you are going too fast!

2. Utilize examples

- Instead of writing all the code from scratch, it is good to look at a few examples.
- Also, majority of the learners prefer to look at examples instead of reading a manual.

3. Use a good IDE

- IDE stands for Integrated Development Environment
- *Examples*: MS Visual Studio, NetBeans, Eclipse, jGRASP, DrJava, BlueJ, …
- Good IDE takes care of mundane things and makes programming enjoyable!

4. Plan before you code

- You can be “slow and steady” or “race and burn”
- It is common for experienced designers to do “race and burn” before reverting back to “slow and steady” ☺️
- Make a practice of writing high level pseudo code before coding – unfortunately, this is not insisted in most programming courses!

5. Learn with a friend

- Learning in a group setting is preferred, if not, try to learn with a friend.
- Discuss ideas and help each other when you get stuck.
- Enables you to work on a team assignment
- Self-paced learning alone is not for every one.
- It requires lot of self-discipline & it is not much fun!

6. Mimic an interesting game/feature

- It is easier to focus on implementation when functionality is clearly understand.
- It is easy to “relate to”.

7. Have a time-discipline

- I encourage you to fix the issues on your own, but do not spend >30 minutes on any one issue. Ask for help!
- If not, your frustration level will increase & confidence will go down.

8. Implement a useful app/game

- It increases your confidence level.
- You can be proud of your work.

9. Use video tutorials

- www.spoken-tutorials.org
- www.khanacademy.org/cs
- …

10. Participate in programming competitions

- ACM programming contest - uva.onlinejudge.org
- Infosys Aspirations 2020
- US Computing Olympiad (feeds to International Olympiad in Informatics) – www.unaco.org

Advanced level

- Problem solving & algorithms
- Programming competitions
- Advanced game development
- Sophisticated Mobile app development

Suggested sequence