Function Basics

Quispe

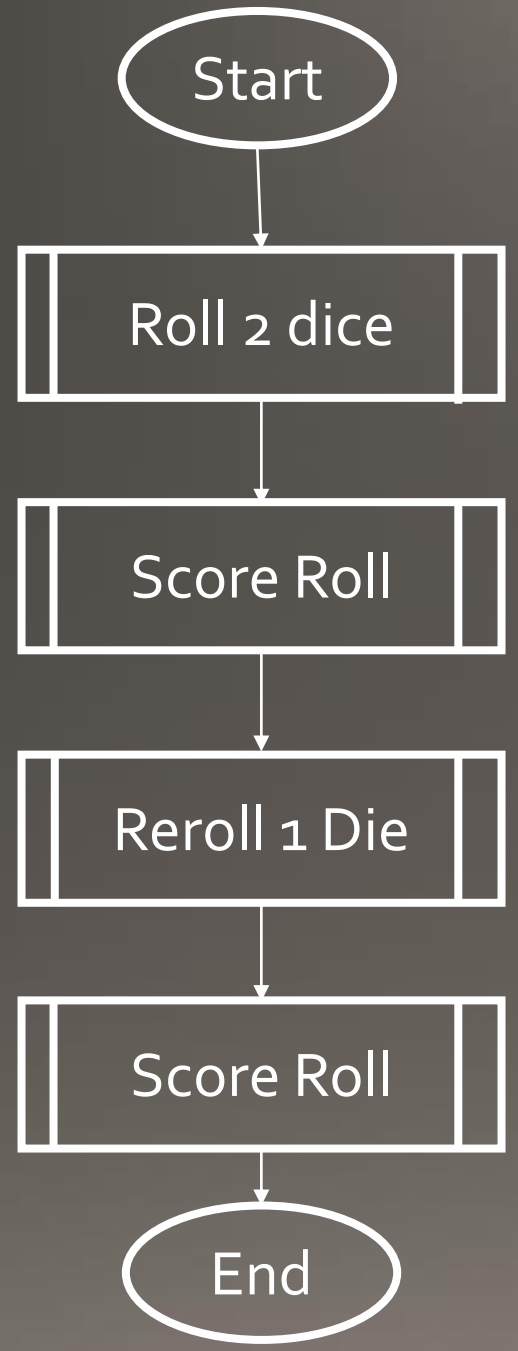
Top-down Design

- Start with a large problem
- Break it down into smaller problems – may become modules in your flow chart
- Repeat until it is clear how to solve each of the smallest problems

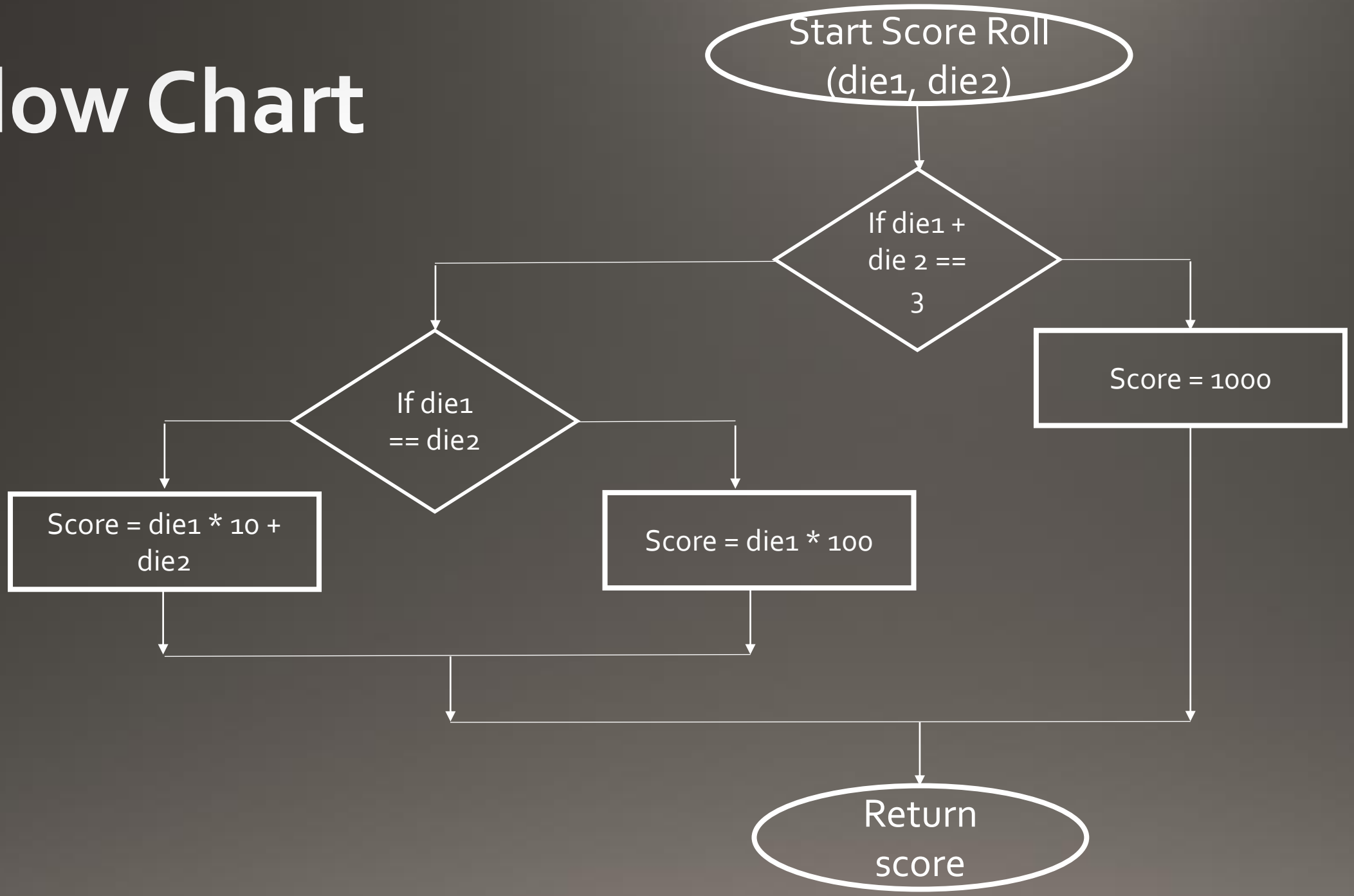
Quispe

- Dice Game
- Roll 1 and 2 to win 1000 points
- Roll doubles to win one die * 100 points
- Basic roll, first die * 10 + the second die

Flow Chart



Flow Chart


Start Score Roll (die1, die2)
If die1 + die 2 == 3
Score = 1000
If die1 == die2
Score = die1 * 10 + die2
Score = die1 * 100
Return score

Code Reusability

- Implemented as functions in Python
- Represent the modules from your flowchart
- May be called multiple times during execution of one program
- May be used in any program where applicable

Information Hiding

- Functions may be seen as black boxes by someone who is using it only (not true for the person writing it)
- You need an interface that tells you how to correctly use the function
- You simply need to know that it works, not how it works

The code

```
def scoreRoll(die1, die2):  #die1 and die2 are parameters
        score = 0
        if(die1 + die2 == 3):
                score = 1000
        elif (die1 == die2):
                score = die1 * 100
        else
                score = die1 * 10 + die2
        return score;
```

Complete example

```
def scoreRoll(int die1, int die2):
        score = 0
        if (die1 + die2 == 3):
                score = 1000
        elif (die1 == die2):
                score = die1 * 100
        else
                score = die1 * 10 + die2
        return score
```

```
def main():
        die1 = 5;
        die2 = 3;
        score = scoreRoll(die1, die2)
        print("Score:  ", score)

main()
```

Quispe

- Complete the game with 1 user and the computer being the “players”. Dice rolled are a pair of standard 6-sided dice