Decomposing Problems

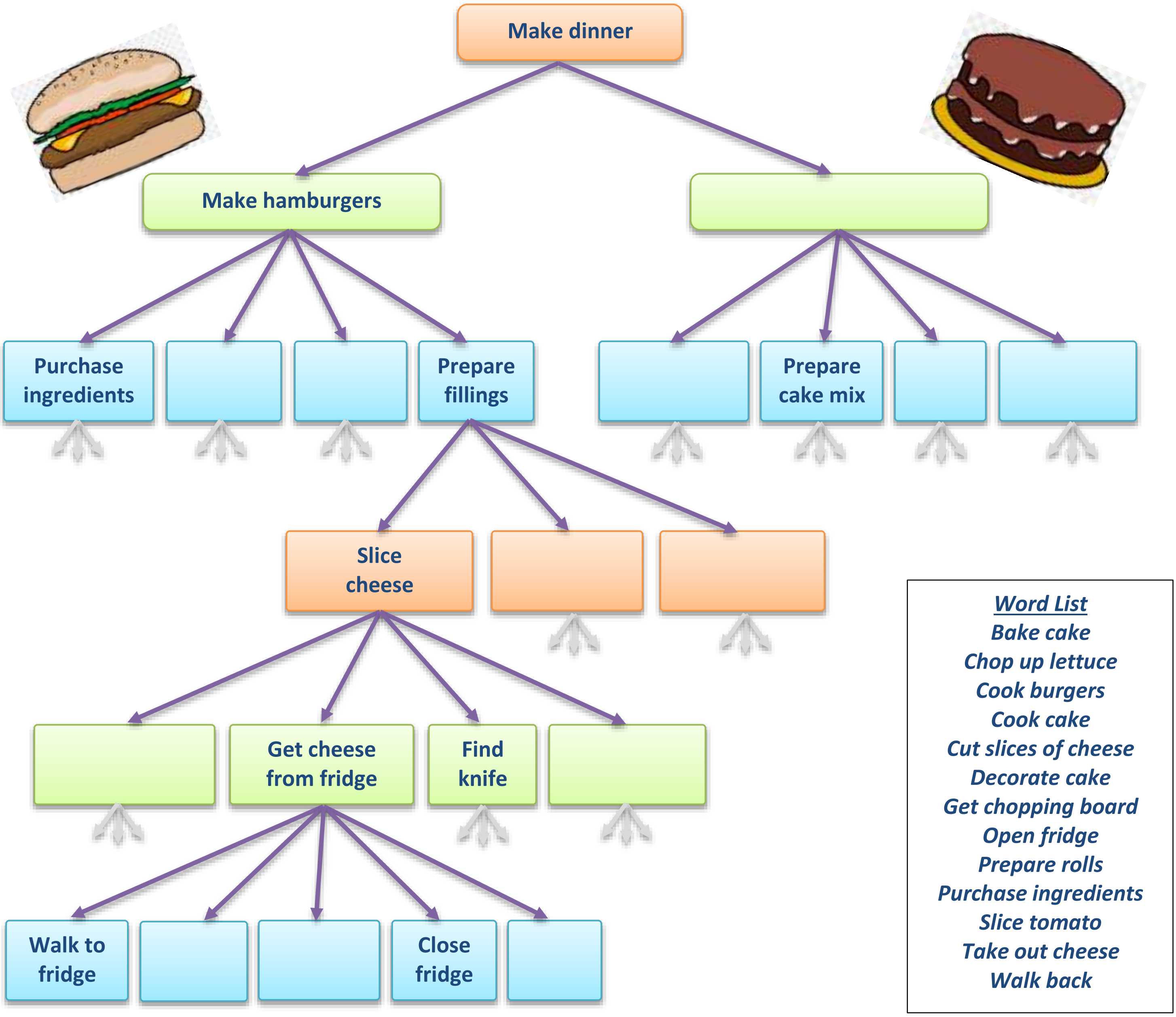
Aim: *To use decomposition, pattern recognition and abstraction when looking at problems.*

As we have seen, decomposition means breaking down a difficult problem into more manageable pieces. A common way to tackle this process is to use a tree diagram showing increasing levels of detail.

Imagine that you want to make dinner for some friends. You decide that you would like to have hamburgers for the main course and a chocolate cake for dessert. We can decompose this problem using a tree diagram.

Task 1 – Decomposing the Problem

The diagram below is only partially complete. Use the terms in the word list to complete the empty boxes.



Word List
Bake cake
Chop up lettuce
Cook burgers
Cook cake
Cut slices of cheese
Decorate cake
Get chopping board
Open fridge
Prepare rolls
Purchase ingredients
Slice tomato
Take out cheese
Walk back

Decomposing Problems (page 2)

Task 2 – Pattern Recognition

We'll now look for sections of the problem that are similar to each other so that we can save time and make our work more efficient. For example, in the diagram above we have 'Purchase ingredients' in two boxes. Obviously we are not going to go to the shop for the hamburger ingredients and then go again separately for the cake ingredients. We would make our work more efficient by visiting the shop only once.

There are lots of boxes missing from our diagram (we have only followed the 'Prepare fillings' through in detail). Think through all the processes involved and write down some things that you could do to save time along the way.

1. *Visiting the shop only once.*
2. ______________________________
3. ______________________________

Task 3 – Abstraction

We have too many details in our tree diagram about how to slice cheese. Everyone knows how to slice cheese, so it's really not necessary to include any more information about this. Similarly, you may decide that you don't need to mention purchasing the ingredients – it goes without saying. Other areas may need more detail, for example how to prepare the cake mix.

Draw another tree diagram only showing what you consider to be the important details. This is called abstraction.

Decomposing Problems (page 3)

Task 4 – Algorithms

An algorithm provides a step-by-step solution to one part of the problem. For example, the recipe for actually baking the cake will be an algorithm. So will be the instructions for the shopping trip.

The instructions for baking a cake are shown below, **but they are not in the correct order**.

a. *Add eggs one at a time to the butter/sugar mixture, beating after each addition.*

b. *Bake for 1 hour 10 minutes.*

c. *Beat butter and sugar until light and fluffy.*

d. *Grease 7cm-deep, 20cm round cake tin.*

e. *Leave the baked cake in the cake tin for 5 minutes.*

f. *Line bases and sides of the greased cake tin with baking paper.*

g. *Start by preheating the oven to 180°C.*

h. *Spread the mixture into prepared tin.*

i. *Stir in flour, cocoa and milk until combined.*

j. *Turn out onto a wire rack to cool completely.*

Suggest the correct order for the instructions. Write the letters only.

Write out some instructions for making the burgers. This is also an *algorithm*.

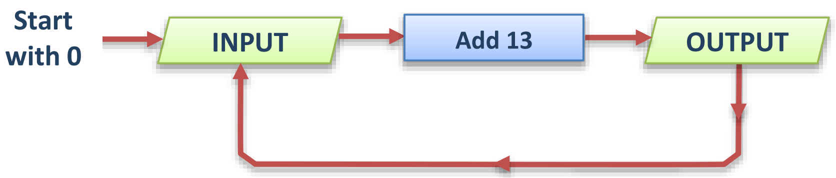
Iteration

Iteration means to repeat a process over and over again. The output (or end result) of one *cycle* becomes the input of the next.

Aim: *To learn about iteration.*

Task 1 – Iteration in Maths

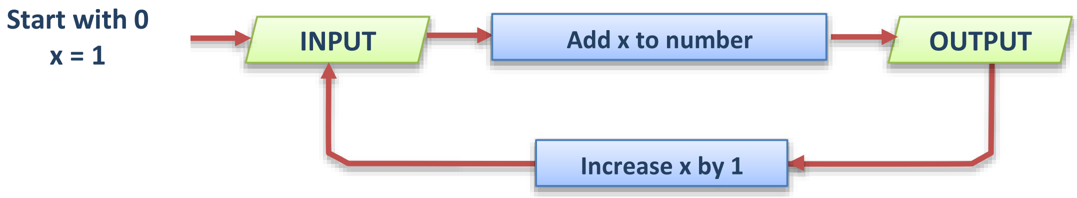
a. Working out your 13 times table involves iteration. This can be shown using the flowchart below.



The whole process is called iteration. Confusingly, each cycle is also called an iteration.

What are the outputs for the first 5 iterations of this process? (The first result is 13) ______________________

b. Here is another iteration where we start with 0, then add 1, then add 2, then add 3 etc.



What are the outputs for the first 10 iterations of this process? ______________________________________

Find out the special name given to this set of numbers ______________________________________

c. Draw a flowchart for an iteration which outputs the following numbers:

1 3 7 15 31

Iteration (page 2)

Task 2 – How Much Salt?

Imagine you are making a big pot of stew. You want to add salt, but not too much (we all know too much salt is bad for you). You might try adding a small pinch of salt, stirring it in, tasting the stew and then doing this again until it is salty enough. Instructions for this part of the recipe might be as follows:

1. Add a small pinch of salt and stir it in.
2. Taste the stew.
3. **IF** the stew is not salty enough, **THEN** add a small pinch of salt and stir it in.
4. Taste the stew.
5. **IF** the stew is not salty enough, **THEN** add a small pinch of salt and stir it in.
6. Taste the stew.
7. ….

The problem is that you don't know how long to go on for. Also, you are writing out the same instructions repeatedly. We can add an iteration to our instructions to make them more efficient.

Try and write a shorter set of instructions for adding the salt. Leave lines 1 and 2 as they are, then add one more line that includes the words **REPEAT** and **UNTIL**.

__NB.__ A cookbook will usually say "Add salt to taste".

Task 3 – White Sauce

Write out some short, efficient instructions for making a white sauce using the information below. Try and cut out all the unnecessary details and use words like **IF**, **THEN**, **REPEAT** and **UNTIL**.

> *First of all, you need to heat some butter in a saucepan – 25g should be enough. Stir the butter while it melts. You should then put in about 25g of plain flour and carry on stirring until you have a nice, smooth mixture. Cook this mixture for a couple of minutes. Now comes the tricky part. You need to take the pan off the heat, add about 50ml of milk, stir it in well, add another 50ml of milk and stir it in well. In fact, keep adding milk 50ml at a time until you have added 600ml. Whatever you do, don't stop stirring whilst you are adding milk or the white sauce will go lumpy. Once all the milk is added and your sauce is nice and smooth, put the saucepan back on the heat and, still stirring all the time, bring it to the boil. Simmer the sauce over a low heat for 8 minutes and then add as much salt as you think it needs. You can also add pepper if you like the taste of it.*

11 Flowcharts

A flowchart can be used to represents an algorithm, showing the steps of the process as different shaped boxes connected by arrows. Flowcharts help you think through a process and make sure that all instructions are in the correct order.

Aim: *To formally introduce the use of flowcharts.*

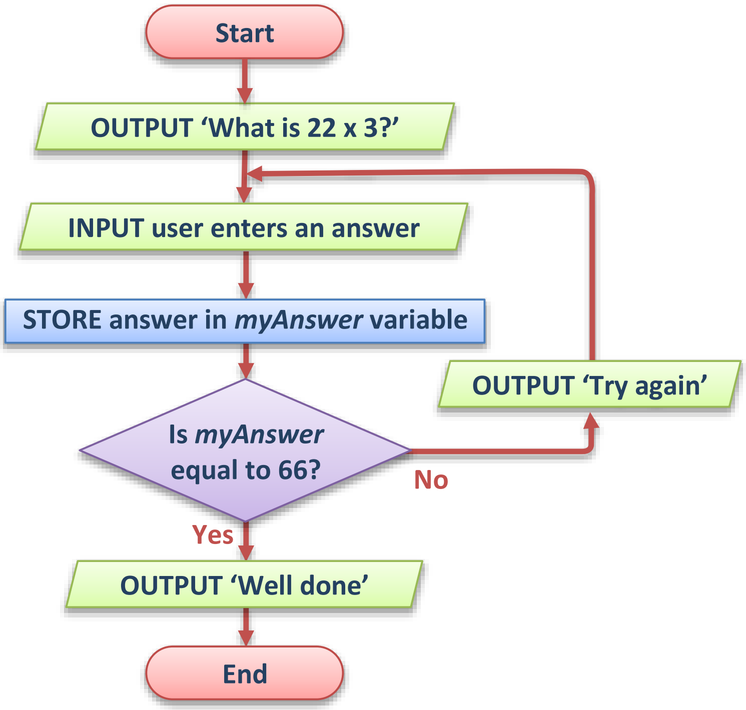
Task 1 – Flowchart Symbols

Draw connectors to match each flowchart symbol to its name and use. Look the answers up on the web if you need help, or try and guess them using the flowchart in the next task. The colours are not significant; the shapes are.

Symbol			Name			Use
▭	•	•	**Input or Output**	•	•	Connects the symbols and shows the direction of the flow
▱	•	•	**Flow Line**	•	•	Shows the start and end of the process
▭	•	•	**Terminal**	•	•	Where data is received or displayed
◇	•	•	**Process**	•	•	A decision, usually a “Yes” or “No”
→	•	•	**Decision**	•	•	An instruction or command

Task 2 – A Simple Flowchart

Here is a flowchart for a simple process. Explain in plain English how the process works.


Start
OUTPUT ‘What is 22 x 3?’
INPUT user enters an answer
STORE answer in *myAnswer* variable
Is *myAnswer* equal to 66?
No
OUTPUT ‘Try again’
Yes
OUTPUT ‘Well done’
End

Flowcharts (page 2)

Task 3 – Pseudocode from a Flowchart

Complete the pseudocode for the flowchart in the previous task.

```
OUTPUT ‘What is 22 x 3?’
REPEAT
  INPUT user inputs their answer
  STORE the input in the myAnswer variable
______________________________________________
______________________________________________
______________________________________________
______________________________________________
______________________________________________
```

Task 4 – Flowchart from Pseudocode

See if you can draw a flowchart for the pseudocode below. We have introduced three new terms:

>=	*this symbol means ‘greater than or equal to’*
<=	*this symbol means ‘less than or equal to’*
AND	*this is a logical operator; both conditions have to be true in the IF statement*

Note: *the flowchart will have two paths to the ‘END’ terminal.*

```
OUTPUT 'How old are you?'
INPUT user inputs their age
STORE the user's input in the age variable
IF age >= 13 AND age <= 19 THEN
    OUTPUT 'You are a teenager'
ELSE
    OUTPUT 'You are not a teenager'
```

Task 5 – Further Flowcharts

You may have previously written algorithms for the following problems. Create a flowchart for each one.

a. Take a number from the user and multiply it by 12345. Display the answer.

b. Take two numbers from the user, store them in two different variables, multiply them together and display the answer.

c. Ask the user a maths question. Display the words “Well done” if they enter the correct answer, otherwise let them try again.

Sorting is a common task in computing. There are lots of reasons why we may want to put data in a particular order. For example, imagine looking through the contacts on your phone if they were not sorted alphabetically. It may also be necessary to keep data sorted so that the computer can use faster search algorithms, such as the *binary search*.

Aim: *To investigate how a sorting algorithm might work.*

Task 1 – The Bubble Sort

A bubble sort algorithm orders data one pair at a time, switching the data if necessary. It compares data items 1 and 2, then 2 and 3, then 3 and 4 etc., swapping the values each time if they are out of order. When it gets to the end of the data, it starts again on another 'pass'. When a pass through the data results in no change, then the data is sorted.

Complete the table below showing what happens to the 5 items of data D, C, E, A, B in a bubble sort. Remember that you can only switch one pair of values in each column (they have been highlighted). You only switch the pair of values if they are not already in order.

Pass 1

	D	C	C	C	C
	C	D	D	D	D
	E	E	E	A	A
	A	A	A	E	B
	B	B	B	B	E
All in order?	N	N	N	N	N
Switch pair?	Y	N	Y	Y	☒
Change in pass?	Y				

Pass 2

	C				
	D				
	A				
	B				
	E				
All in order?	N				
Switch pair?	N				☒
Change in pass?					

Pass 3

All in order?					
Switch pair?					☒
Change in pass?					

Pass 4

					A
					B
					C
					D
					E
All in order?					Y
Switch pair?					☒
Change in pass?					

All in order? *Place a 'Y' in this box if all items are sorted, else enter an 'N'.*
Switch pair? *Place a 'Y' in this box if the highlighted pair need to be switched, else enter an 'N'.*
Change in pass? *Place a 'Y' in this box if there are any switches in the whole pass, else enter an 'N'.*

4. How many pair comparisons are made in each pass? ____________________

5. If 'A' started as the last item, how many passes before the data must be sorted? ____________________

6. What is the maximum number of pair comparisons required for a list of 5 items? ____________________

7. If a list had 1 thousand items, how many comparisons might be needed to sort the data? ____________________

A bubble sort is a relatively simple algorithm, but it can be very slow to sort large lists of data.

Sorting (page 2)

Task 2 – The Bucket Sort

A bucket sort is a 3 step process. Let's imagine we have 200 test papers with names on and we want to sort them into alphabetical order. The 3 steps might be as follows.

1. Place each paper into a group. For example, you may have a 'bucket' for each letter of the alphabet.
2. Order the papers in each group using some other sorting method (e.g. the bubble sort).
3. Collect together the sorted papers from each group to form a complete, ordered set.

Complete the bucket sort below. Go through each name in the first column and add it to the correct bucket in the middle column. Following this, look at the contents of each bucket in turn, sort the names in your head and then copy them in order into the last column.

Original Data
Frank
Emily
Shelly
Kechira
Dollie
Natalie
Tashi
Sandra
Serena
Dzung
Sina
Bede
George
Maria
Helen
Liz
Georgina
Javed
Isabelle
Adela
Tuan
Madison

Bucket A - E
Emily
Dollie

Bucket F - H
Frank

Bucket I - L
Kechira

Bucket M - R

Bucket S - Z
Shelly

Sorted Data
Adela
Bede

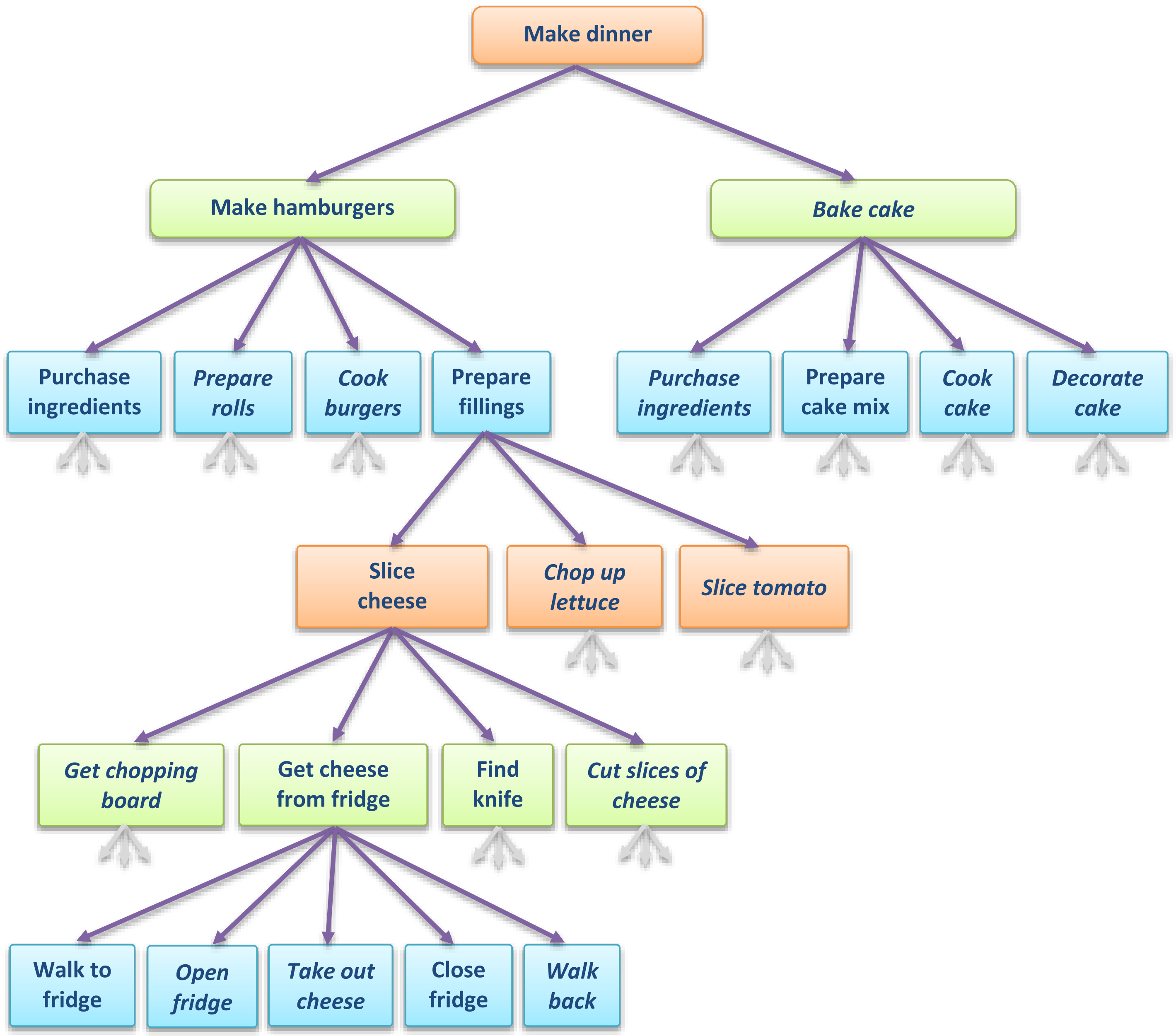
Extension

If we had 1 thousand random names, have a go at working out the number of operations that might be required to sort the data. Think about how many buckets you would use, how many items may go into the average bucket and how many comparisons would be required to sort each bucket using a bubble sort. Moving an item into or out of a bucket can also be considered 1 operation. How does your result compare to using only the bubble sort?

Decomposing Problems - ANSWERS

Task 1 – Decomposing the Problem

Order of boxes may vary across a line.



Make dinner

Make hamburgers

Bake cake

Purchase ingredients

Prepare rolls

Cook burgers

Prepare fillings

Purchase ingredients

Prepare cake mix

Cook cake

Decorate cake

Slice cheese

Chop up lettuce

Slice tomato

Get chopping board

Get cheese from fridge

Find knife

Cut slices of cheese

Walk to fridge

Open fridge

Take out cheese

Close fridge

Walk back

Decomposing Problems (page 2) - ANSWERS

Task 2 – Pattern Recognition

1. Visiting the shop only once.

2. E.g. get everything out of the fridge in one go

3. E.g. use the chopping board and knife to slice everything that needs slicing

Decomposing Problems (page 3)

Task 4 – Algorithms

a. *Add eggs one at a time to the butter/sugar mixture, beating after each addition.*

b. *Bake for 1 hour 10 minutes.*

c. *Beat butter and sugar until light and fluffy.*

d. *Grease 7cm-deep, 20cm round cake tin.*

e. *Leave the baked cake in the cake tin for 5 minutes.*

f. *Line bases and sides of the greased cake tin with baking paper.*

g. *Start by preheating the oven to 180°C.*

h. *Spread the mixture into prepared tin.*

i. *Stir in flour, cocoa and milk until combined.*

j. *Turn out onto a wire rack to cool completely.*

Suggest the correct order for the instructions. Write the letters only.

g, d, f, c, a, i, h, b, e, j

Iteration - ANSWERS

Task 1 – Iteration in Maths

a. Working out your 13 times table involves iteration.

What are the outputs for the first 5 iterations of this process? (The first result is 13) ***13, 26, 39, 52, 65***

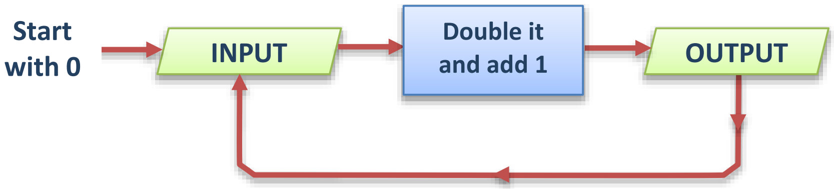
b. Here is another iteration where we start with 0, then add 1, then add 2, then add 3 etc.

What are the outputs for the first 10 iterations of this process? ***1, 3, 6, 10, 15, 21, 28, 36, 45, 55***

Find out the special name given to these numbers. ***Triangle numbers***

c. Draw a flowchart for an iteration which outputs the following numbers:

1 3 7 15 31



Task 2 – How Much Salt?

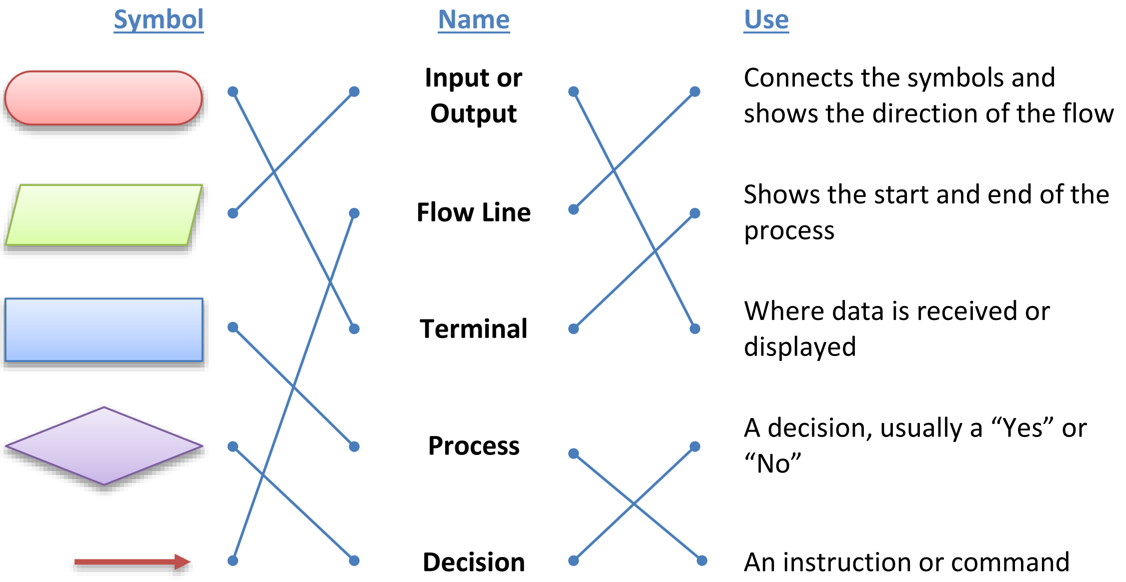
1. Add a small pinch of salt and stir it in.
2. Taste the stew.
3. **REPEAT** steps 1 & 2 **UNTIL** the stew is salty enough.

Task 3 – White Sauce

1. *Melt 25g butter in a saucepan.*
2. *Add 25g of plain flour and stir until the mixture is smooth. Cook for 2 minutes.*
3. *Remove pan from the heat.*
4. *Add 50ml of milk, stir in well.*
5. *Repeat step 4 until 600ml of milk has been added, stirring all the time.*
6. *Return to the heat and, still stirring, bring to the boil.*
7. *Simmer the sauce over a low heat for 8 minutes and add salt to taste.*
8. *If you like pepper then add some.*

Flowcharts - ANSWERS

Task 1 – Flowchart Symbols


Symbol
Name
Use
Input or Output
Flow Line
Terminal
Process
Decision
Connects the symbols and shows the direction of the flow
Shows the start and end of the process
Where data is received or displayed
A decision, usually a "Yes" or "No"
An instruction or command

Task 2 – A Simple Flowchart

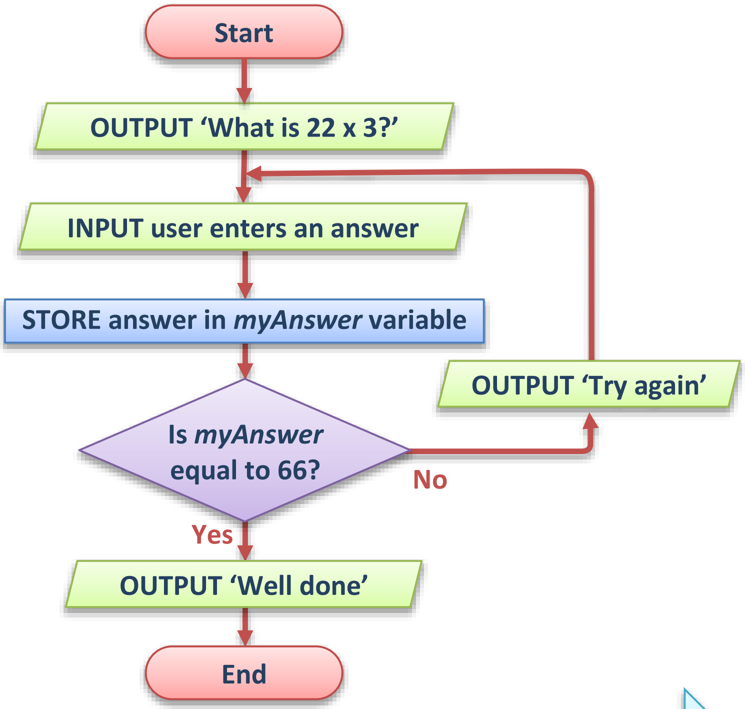
Here is a flowchart for a simple process. Explain in plain English how the process works.

Displays the question 'What is 22 x 3?'.

If the user enters the answer 66

then the text 'Well done' is displayed.

Else the text 'Try again' is displayed.


Start
OUTPUT 'What is 22 x 3?'
INPUT user enters an answer
STORE answer in *myAnswer* variable
Is *myAnswer* equal to 66?
No
OUTPUT 'Try again'
Yes
OUTPUT 'Well done'
End

Flowcharts (page 2) - ANSWERS

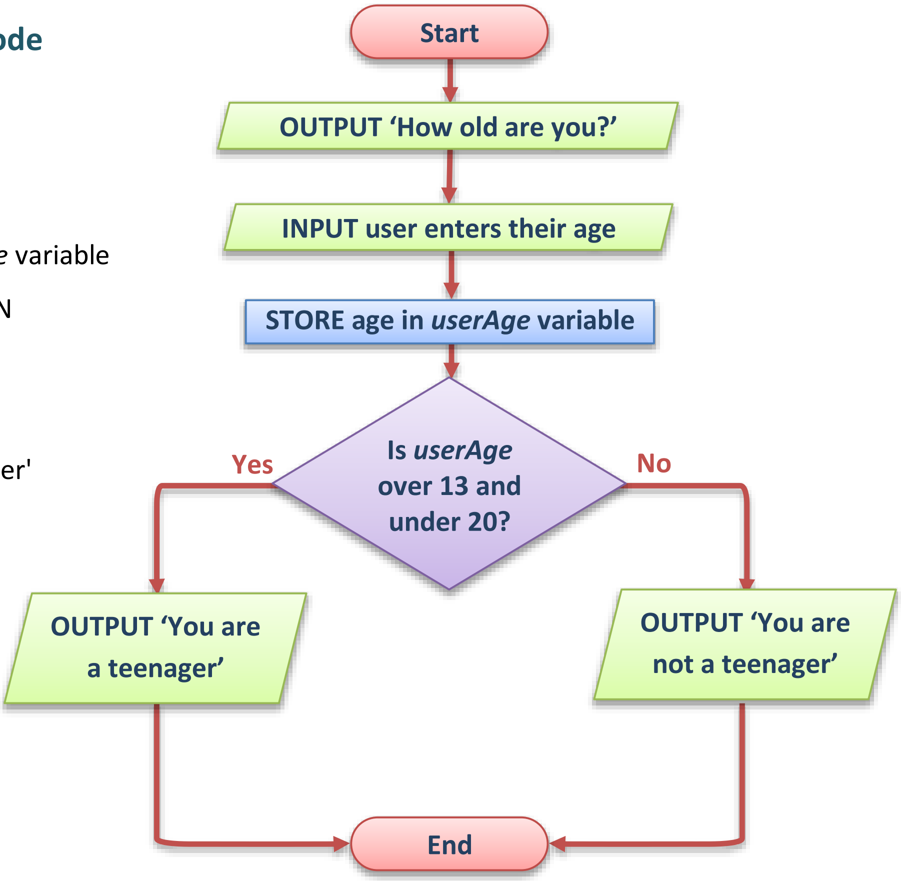
Task 3 – Pseudocode from a Flowchart

Complete the pseudocode for the flowchart in the previous task.

```
OUTPUT ‘What is 22 x 3?’
REPEAT
  INPUT user inputs their answer
  STORE the input in the myAnswer variable
    IF myAnswer = 66 THEN
      OUTPUT ‘Well done’
    ELSE
      OUTPUT ‘Try again’
UNTIL myAnswer = 66
```

Task 4 – Flowchart from Pseudocode

```
OUTPUT 'How old are you?'
INPUT user inputs their age
STORE the user's input in the age variable
IF age >= 13 AND age <= 19 THEN
    OUTPUT 'You are a teenager'
ELSE
    OUTPUT 'You are not a teenager'
```

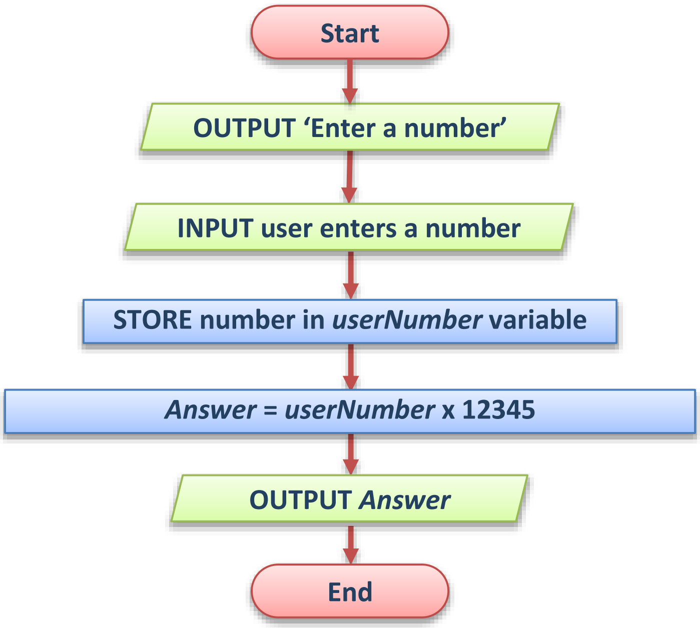

Start
OUTPUT ‘How old are you?’
INPUT user enters their age
STORE age in *userAge* variable
Is *userAge* over 13 and under 20?
Yes
No
OUTPUT ‘You are a teenager’
OUTPUT ‘You are not a teenager’
End

Flowcharts (page 2 cont) – ANSWERS

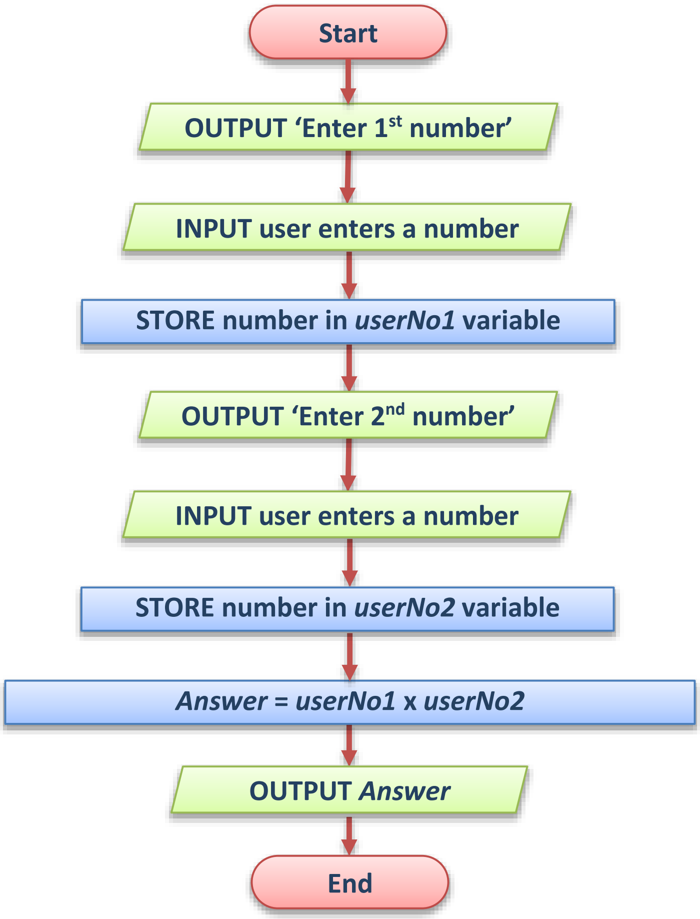
Task 5 – Further Flowcharts

Solutions may vary

a. Take a number from the user and multiply it by 12345. Display the answer.


Start
OUTPUT 'Enter a number'
INPUT user enters a number
STORE number in *userNumber* variable
Answer = *userNumber* x 12345
OUTPUT *Answer*
End

b. Take two numbers from the user, store them in two different variables, multiply them together and display the answer.


Start
OUTPUT 'Enter 1st number'
INPUT user enters a number
STORE number in *userNo1* variable
OUTPUT 'Enter 2nd number'
INPUT user enters a number
STORE number in *userNo2* variable
Answer = *userNo1* x *userNo2*
OUTPUT *Answer*
End

c. Ask the user a maths question. Display the words "Well done" if they enter the correct answer, otherwise let them try again.

As Task 2

Task 1 – The Bubble Sort

	Pass 1					Pass 2					Pass 3					Pass 4				
	D	C	C	C	C	C	C	C	C	C	C	A	A	A	A	A	A	A	A	A
	C	D	D	D	D	D	D	A	A	A	A	C	B	B	B	B	B	B	B	B
	E	E	E	A	A	A	A	D	B	B	B	B	C	C	C	C	C	C	C	C
	A	A	A	E	B	B	B	B	D	D	D	D	D	D	D	D	D	D	D	D
	B	B	B	B	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E
All in order?	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y
Switch pair?	Y	N	Y	Y	☒	N	Y	Y	N	☒	Y	Y	N	N	☒	N	N	N	N	☒
Change in pass?	Y					Y					Y					N				

1. How many pair comparisons are made in each pass? ***4***

2. If 'A' started as the last item, how many passes before the data must be sorted? ***4***

   Note – you do not need an extra test pass as you are already doing the maximum required.

3. What is the maximum number of pair comparisons required for a list of 5 items? ***16***

4. If a list had 1 thousand items, how many comparisons might be needed to sort the data?

   999 x 999 = 998,001

Sorting (page 2) - ANSWERS

Task 2 – The Bucket Sort

Original Data
Frank
Emily
Shelly
Kechira
Dollie
Natalie
Tashi
Sandra
Serena
Dzung
Sina
Bede
George
Maria
Helen
Liz
Georgina
Javed
Isabelle
Adela
Tuan
Madison

Bucket A-E
Emily
Bede
Dollie
Dzung
Adela

Bucket F-H
Frank
George
Helen
Georgina

Bucket I-L
Kechira
Liz
Javed
Isabelle

Bucket M-R
Natalie
Maria
Madison

Bucket S-Z
Tashi
Shelly
Sandra
Serena
Sina
Tuan

Sorted Data
Adela
Bede
Dollie
Dzung
Emily
Frank
George
Georgina
Helen
Isabelle
Javed
Kechira
Liz
Madison
Maria
Natalie
Sandra
Serena
Shelly
Sina
Tashi
Tuan

Sorting (page 2 cont) - ANSWERS

Extension

If we had 1 thousand random names, have a go at working out the number of operations that might be required to sort the data. Think about how many buckets you would use, how many items may go in the average bucket and how many comparisons would be required to sort each bucket using a bubble sort. Moving an item into or out of a bucket can also be considered 1 operation. How does your result compare to using only the bubble sort?

Possible answer

1000 names
26 buckets = 1 for each letter of the alphabet
38.4 names on average per bucket
Say 38 names in 14 buckets and 39 names in 12 buckets

38 operations moving names in to bucket
38 names = 37 x 37 = 1369 maximum comparisons using bubble sort
38 operations moving names out of bucket
=1445 maximum operations

x14 buckets = 20,230 maximum operations

39 operations moving names in to bucket
39 names = 38 x 38 = 1444 maximum comparisons using bubble sort
39 operations moving names out of bucket
=1522 maximum operations

x12 buckets = 18,264 maximum operations

TOTAL = 38,494 maximum operations

Answer will vary depending on actual make up of list

Compared to 998,001 with the pure bubble sort