Queen Mary
University of London

Punch Card Searching

(Companion to the *Australian Magician's Dream* Activity)

Age group: 11 – adult

Abilities assumed: None

Time: 10-15 minutes + 20 minutes to do magic trick grab.

Size of group: 1 upwards

Focus

Search algorithms
Binary Numbers
Divide and Conquer
Computational Thinking: translating problems between domains

Syllabus Links

This activity can be used (for example)

- as a general introduction to computing topics from KS3 up.
- to introduce computational thinking problem solving from KS3 up,
- to teach specific syllabus topics such as:
  KS3: understand several key algorithms (searching) that reflect computational thinking; use logical reasoning to compare the utility of alternative algorithms for the same problem

Summary

Demonstrate how early computers were able to find data stored on punch cards and show that it is the same algorithm as the magic trick in the 'Australian Magician's Dream' Activity. It aims to show in a visual way how a search algorithm works. In doing so you demonstrate a practical use of binary numbers, a divide and conquer algorithm for searching and explore the computational thinking trick of translating solutions between problem areas.

Technical Terms

Search algorithm, divide and conquer, binary numbers, computational thinking.

Materials

Set of at least 24 punch cards made from A4 card.
A pencil
Slides, handouts or similar illustrating binary number representations
(see linked slides at www.teachinglondoncomputing.org)

What to do

Set-up:
Create a set of punch cards by printing those given at the end of this sheet. To print without margins you will need to choose as the paper size "borderless A4" if your printer allows it. If not guillotine the margins. Ideally print them on to thin card. Then cut out the notches / holes as indicated by the dotted lines. Sprinkling talcum powder on the cards can help stop them sticking together during the activity.

The Grab:
The grab for this activity is to do the Australian Magician's Dream magic trick available from Teaching London Computing (www.teachinglondoncomputing.org). Do that magic trick. Then announce that the trick was used by the first computers to find data stored on punch cards.

The activity:
Take the pile of punch cards and mix them up so they are in a random order. Explain how you can find any card using the Down-Under Deal from the magic trick. You will use the same algorithm. To do it you just need to put numbers on the cards using a special code of holes and slots.

To make it work you need to know some simple **binary** maths. Binary is one way numbers are stored in computers and it is just the same idea as the code of holes and slots needed to find things using the trick.

Explain that binary is just a different way of writing numbers. It is a special code, where we are only allowed to use the digits 0 and 1 rather than digits 0-9 as we normally do. We use Base 10. Binary is just Base 2. The base tells us how many different digits – different symbols – that we have available. In base 10 we use the digits to count up to 9 but then run out of digits so have to use a new column. We go back to 0 but carry 1 in to the next column where it stands for 10. In Base 10 the number 16 is 1 lot of 10 (a 1 in the 10s column) and then 6 units. We add 10 and 6 to get the number 16. Similarly 987 means 9 lots of 100, 8 lots of 10 and 7 1s added together.

	100	10	1
x	9	8	7
=	900 +	80 +	7

Binary works in just the same way except that we run out of digits and have to use a new column when we get to 1, rather than 9. Instead of 1s, 10, and 100s, that means the columns stand for 1s, 2s, 4s, 8s and so on. So we write the number 5 in binary (so using only digits 1 and 0) as 101. It is 1 lot of 4 plus 0 lots of 2 plus 1 unit.

5 in Binary is 00101.

	16	8	4	2	1
x	0	0	1	0	1
=	0 +	0 +	4 +	0 +	1

Similarly, 16 in Binary is 10000.

	16	8	4	2	1
x	1	0	0	0	0
=	16 +	0 +	0 +	0 +	0

What does this have to do with punch cards? (Show the punch cards while explaining). Well we can put binary numbers onto cards using holes for 0 and slots for 1. To put the number 5 onto a punch card, starting from the left, we need a hole, then another hole, then a slot, a hole and finally a slot. For the number 16 we need a slot then the rest holes. With space for 5 holes we can put any number up to 31 on a card. With enough space we can put *any* number onto a punch card like this though.

Once we have put the number of a card onto it in binary as holes and slots we can easily find any individual card we want. We just do a version of the Down-under deal, working through the binary of the number we want to find. As we do this:

0 means DISCARD the "down" pile
1 means KEEP the "down" pile

Take the stack of cards with the holes lined up and put a pencil through each position in turn, shaking out all the cards with a slot in that position before moving on. Starting with the pencil in the rightmost position (the units column) and working through the binary digits shaking out cards you can be left with any card required. You use the above code: if there is a 0 in the binary at that position of the number you are looking for then *discard* the 'down' pile – the cards that shake out. If there is a 1 in that position in the target number then *keep* the down pile.

Let's look at an example. (Go through the steps with the pile of cards as you explain this). Let's find card 16. In binary it is 10000. From the right that becomes

0: DISCARD those that fall
0: DISCARD those that fall
0: DISCARD those that fall
0: DISCARD those that fall
1: KEEP those that fall.

We repeatedly discard the down pile until on the 5th round we keep the down card. This is exactly what we did in the trick! Let's find card 5. In binary it is 00101. From the right that becomes

1: KEEP those that fall
0: DISCARD those that fall
1: KEEP those that fall
0: DISCARD those that fall
0: DISCARD those that fall

We will be left with card 5. By spelling out the binary of its number in this way we can find any card we want!

The explanation:

This can just be used to visually demonstrate a search algorithm. There are a variety of further directions explanations can go, however for a deeper session as we now discuss.

For a short, simple presentation of why the algorithm is the same as in the magic trick just demonstrate that they both remove the same cards in the first round by doing it and looking at the cards, and noting that each round similarly removes every second card.

The following is intended as a longer explanation if you wish to have and/or give a deeper understanding.

Think about what is happening as you shake out the cards. It is exactly the same thing happening as the down under deal. Take the first round of discarding cards. Shaking out the cards discards all cards that have a slot (1) in the first position of the binary number. That is the unit column. The numbers 1, 3, 5, 7 and so on have a slot (a 1) in that position – all the odd numbers. That is just because that bit is flipped every second number as we count 00, 01, 10, 11. The last position counts 0, 1, 0, 1 and so on. That is no different to base 10 where we have more digits so count 0, 1,… 9, 0, 1,…9 etc. To think of it another way, as you add up the contribution to the number of the separate digits (like 5= 4 + 0 + 1), it is that last digit that gives a way of making odd numbers as all the other digits represent even numbers!

Having taken out all the odd numbered cards we move on to the next hole on the punch cards (i.e., the next position in the binary number). It shakes out all the numbers that include 2 in the addition that makes up the number. For example 6 is 110 in binary because 6 = 4+2+0. The numbers dropping out this time are 2, 3, 6, 7, 10, 11… The odd ones have already been removed though, leaving 2, 6, 10, … as the cards actually shaken out. This is the same sequence of cards as are removed in the second round of the down-under deal. This is actually every second card that is left. Again if you think of the way the binary counting system works, the second column counts 0, 1, 0, 1, 0, 1 just like the first. However the second digit only changes each time the first binary digit has itself counted 0, 1. This can be seen in the following sequence for three digit binary numbers.

0 0 0

0 0 1

0 1 0

0 1 1

1 0 0

1 0 1

1 1 0

1 1 1

The same thing happens on each round of removing cards. We are actually shaking out every second card that is left. The difference to the trick is that the punch cards are shuffled. The numbers don't refer to the card's position but to its binary hole and slot label. Another difference is that with the punch cards all the cards are removed in parallel –all in one go. Whereas the down-under deal was very slow and boring as we went through every card in turn, the punch card version is very fast. The trick and punch card versions could be used to illustrate the difference between a sequential algorithm and a parallel algorithm if desired.

The algorithm uses **divide and conquer** to solve the problem of finding a card quickly. On each round we do an operation that removes half of the cards that remain. How do we search those remaining? We do the same again, just on a different position of the binary number. Compare this with the algorithm where we check each card in turn to see if it is the one we are looking for (linear search). The divide and conquer algorithm is much faster, precisely because it cuts the size of the problem in half on every step.

In essence, we have show that a magic trick and a method for searching actually use the same underlying algorithm. The solution to one problem (that of entertaining an audience with a magical effect) is a solution to an apparently different problem (that of finding a card in a pile of many). This is an illustration of an important part of **computational thinking**: that of **transforming problems between problem areas**. Someone who knew the trick could use it as the basis of solving a search problem. Similarly, a magician who knew the search algorithm might invent some variation on the trick themselves from it. In fact, a whole series of magic tricks are inspired by algorithms in this way.

We can actually use the understanding we have gained from the search algorithm to come up with new versions of the magic trick, using exactly this computational thinking skill. We have seen that we can find any punch card by discarding and keeping the cards shaken out. We could develop a variation of the trick based on this observation where we end up predicting a card that was originally placed at any position in the pack, not just the 16th. Computational thinking leads to our creating a new magic trick.

Variations and Extensions

Make a set of punch cards
Have the class make their own set of punch cards, and work out where to cut the slots on each card themselves. Blank punch cards are provided at the end.

Student binary numbers
To demonstrate binary numbers, spell out the numbers on a row of students. This could be done by students holding cards with 1s and 0s as well as the amount a 1 stands for in that position (1,2,4,8, etc). Alternatively have a student standing represent 1 and sitting (curled in a ball) represent 0. See the CS Unplugged binary number activity (www.csunplugged.org) for ideas along these lines of how to introduce binary.

Punch card sorting (more challenging)
Challenge the class to come up with a series of down under style deals, removing and replacing punch cards in the pack, that guarantees that the pack will be sorted into numerical order at the end.

Further Reading

The Magic of Computer Science

> *There are lots more magic tricks with computer science twists available from http://www.cs4fn.org/magic/ including several free magic books.*

Links to other activities

The following activities are also available via teachinglondoncomputing.org

The Australian Magician's Dream

Do a magic trick where your predict a card chosen that even the person choosing couldn't have known. Challenge the audience to work out how it is done, teach them how to do the trick and then use it to explain algorithms, searching, and logical reasoning.

20 Questions

Play 20 questions and show you can do divide and conquer problem solving. This introduces the idea of divide and conquer problem solving in the context of search algorithms. It also introduces the idea of efficiency analysis as a way of comparing algorithms.

Locked-in

Explore the design of an algorithm to allow someone who is totally paralysed to communicate.
This gives an introduction to computational thinking based problem solving, leading to an understanding of what an algorithm is and how algorithms can be compared on the basis of efficiency. It also illustrates how computational thinking is about more than just creating computer-based solutions. Computing is about solving problems for people.

Live demonstration of this activity

Teaching London Computing give live sessions for teachers demonstrating this and our other activities. See http://teachinglondoncomputing.org/ for details. Videos of some activities are also available or in preparation.

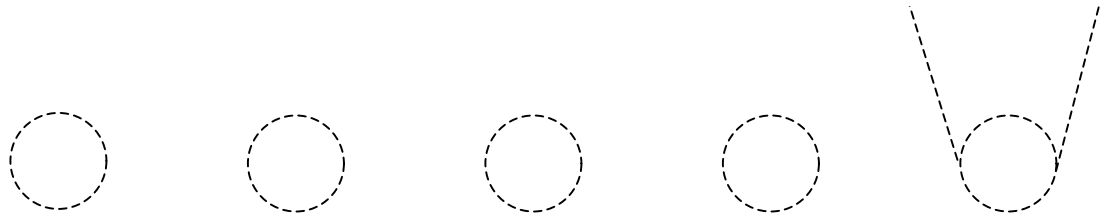
Department for Education
SUPPORTED BY MAYOR OF LONDON
COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE

Queen Mary
University of London

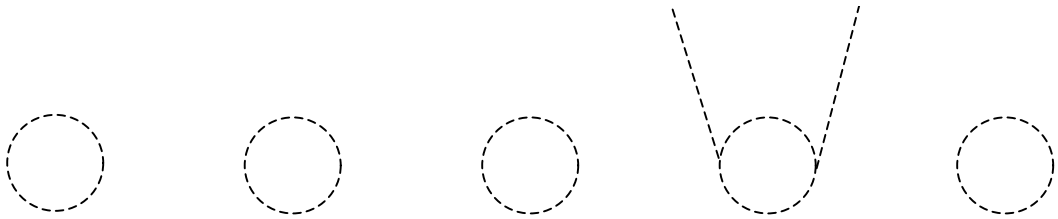
Punch cards

The following cards are ready-made punch cards that just need printing and the notches cutting out.

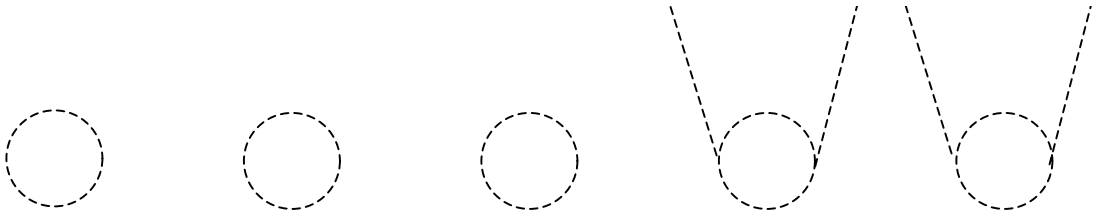
1. Print the following sheets on to thin card
    - To print without margins you will need to choose as the paper size “borderless A4” if your printer allows it.
    - If not guillotine the margins.
2. Cut out the notches / holes as indicated by the dotted lines.
3. Cut off the corner as indicated by the dotted lines
    - This is used to ensure that all the cards are the right way round before use.
4. Sprinkle talcum powder on the cards to help stop them sticking together.



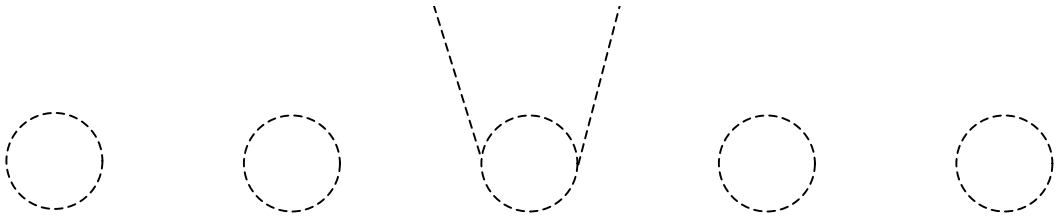
1



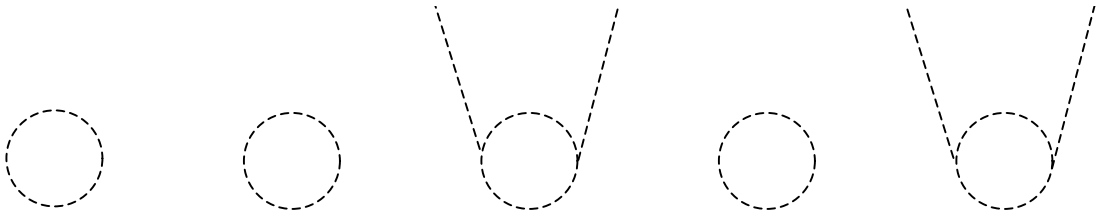
2



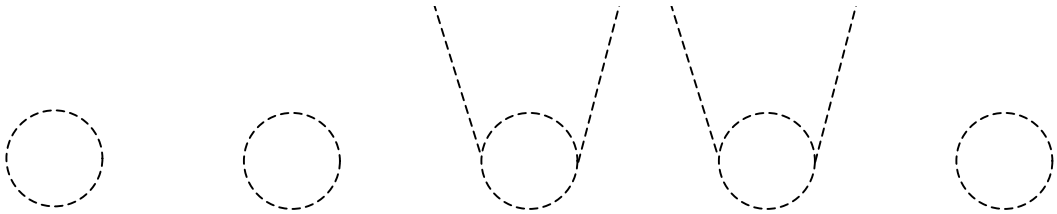
3



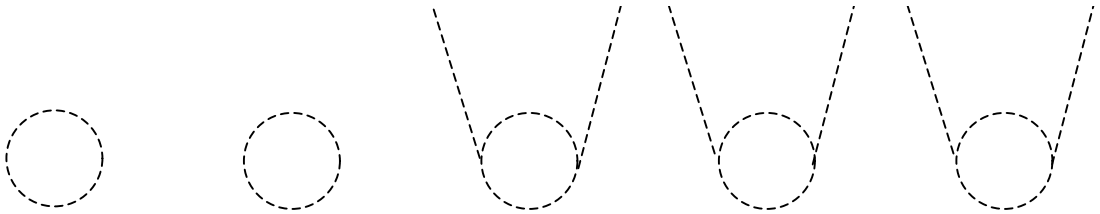
4



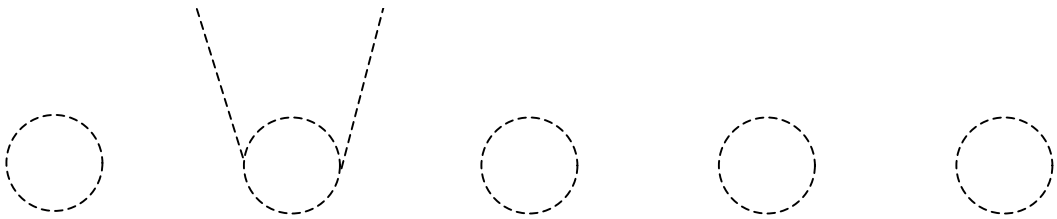
5



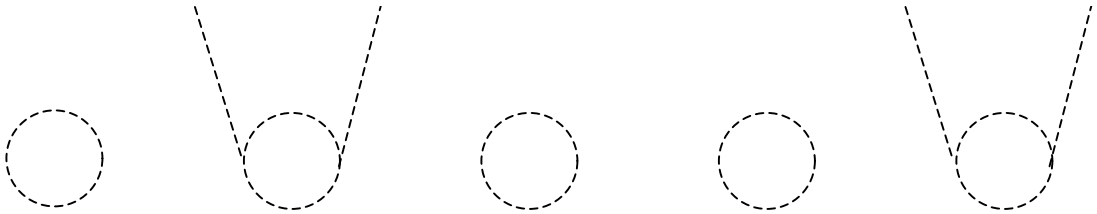
6



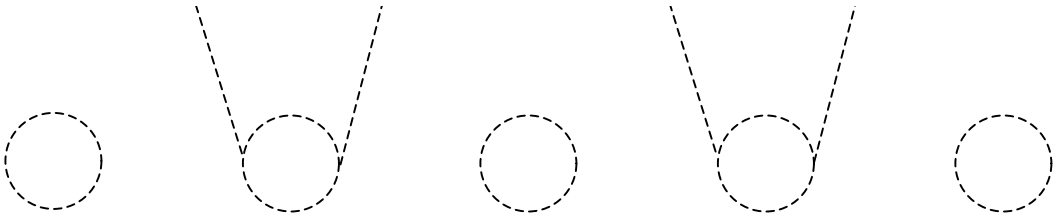
7



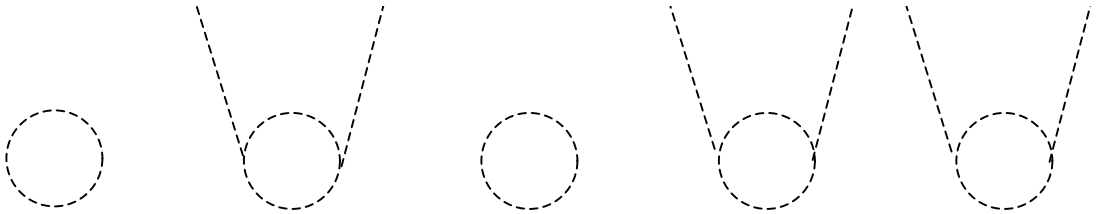
8



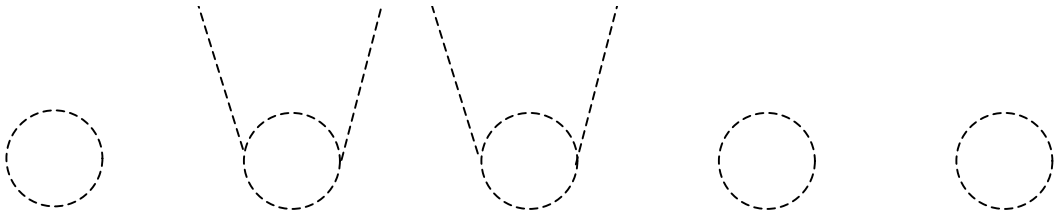
9



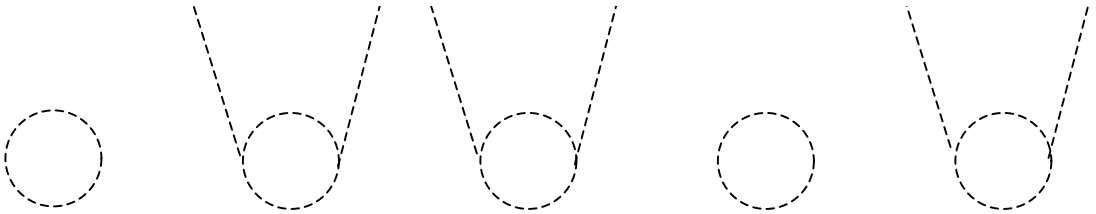
10



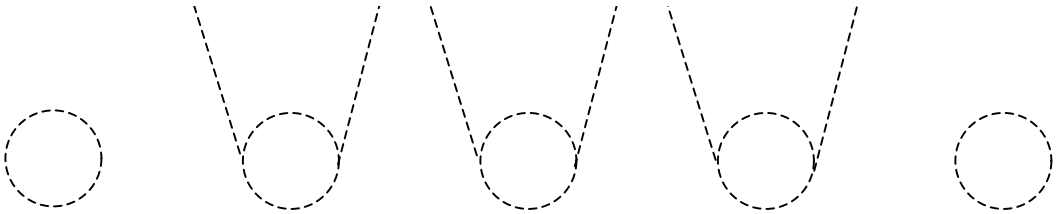
11



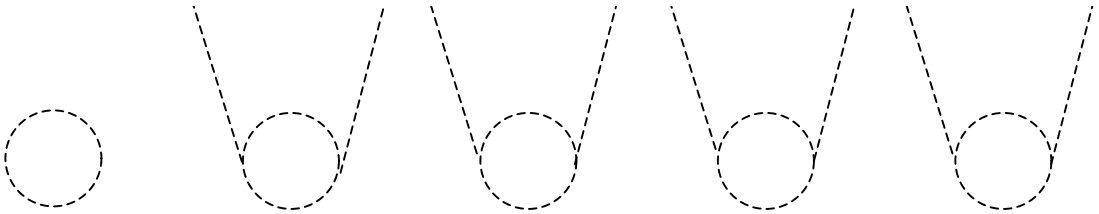
12



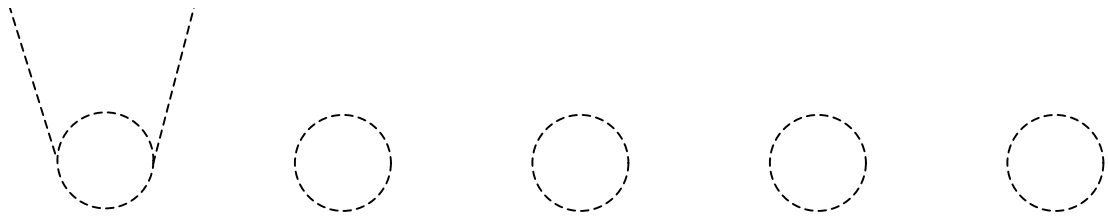
13



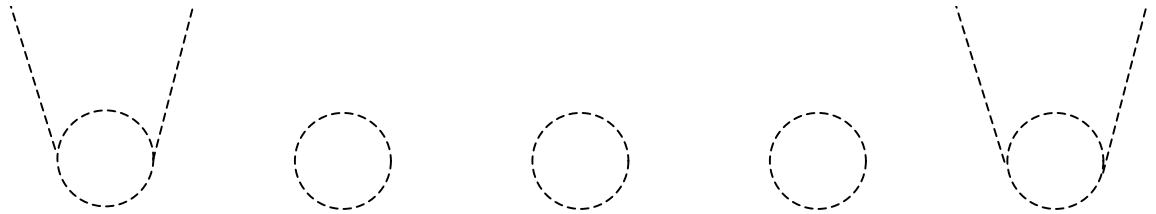
14



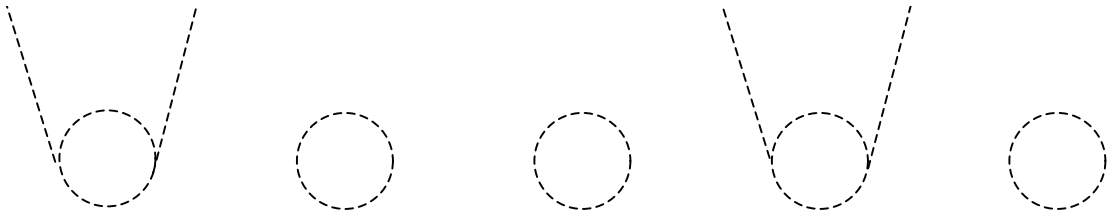
15



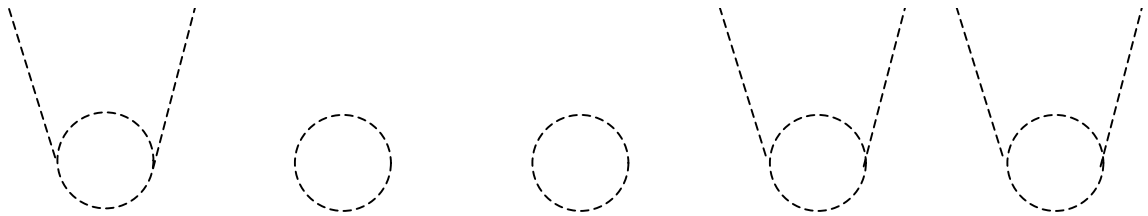
16



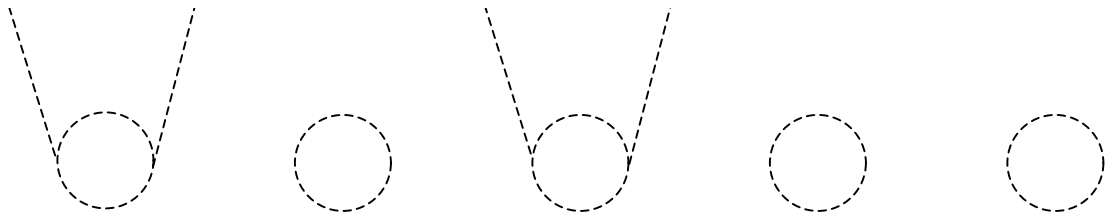
17



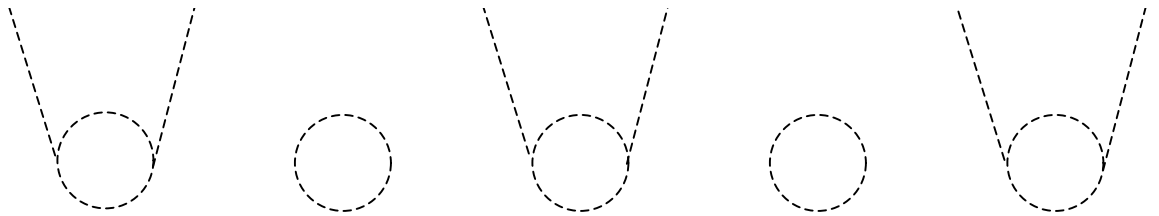
18



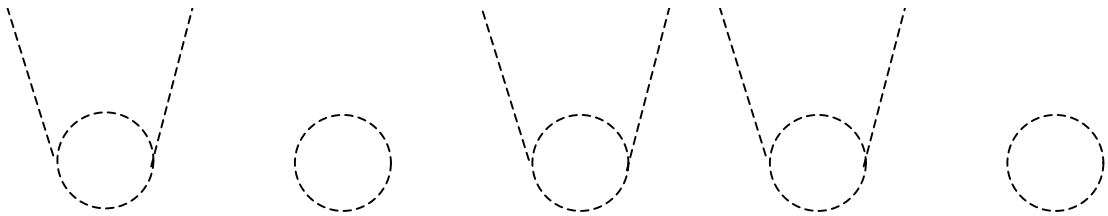
19



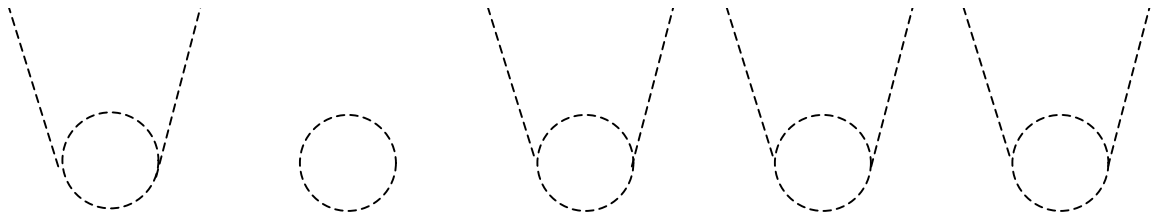
20



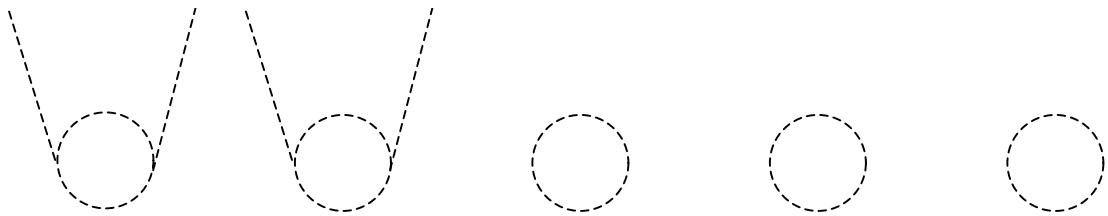
21



22



23



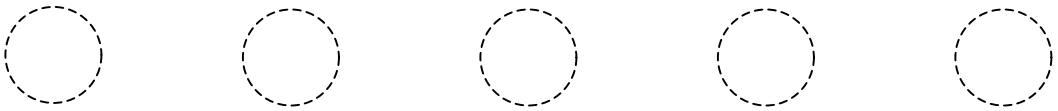
24

Queen Mary
University of London

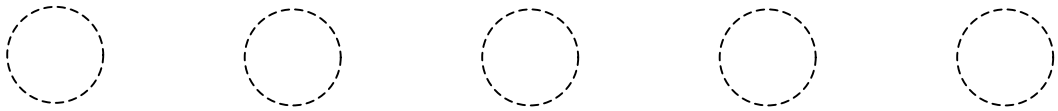
Punch cards:
binary to be added

The following cards are DIY punch cards to allow students to work out the binary and make the punch cards themselves.

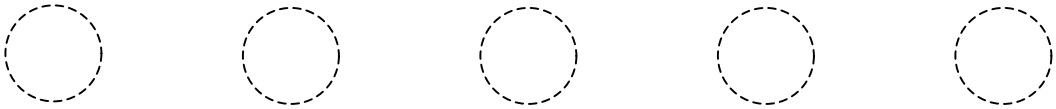
1. Print the following sheets on to thin card
    - To print without margins you will need to choose as the paper size “borderless A4” if your printer allows it.
    - If not guillotine the margins.
2. Draw dotted lines from the holes on each card to make slots where the 1s in the binary should be for that card’s number.
    - For example, 5 is 00101 in binary so should have notches drawn at the points on the first and third holes from the right.
    - Check the notches are drawn in the right place before cutting.
3. Cut out the notches / holes as indicated by the dotted lines.
5. Cut off the corner as indicated by the dotted lines
    - This is used to ensure that all the cards are the right way round before use.
4. Sprinkle talcum powder on the cards to help stop them sticking together.



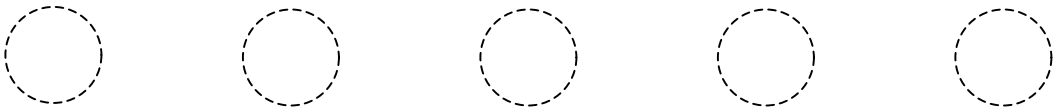
1



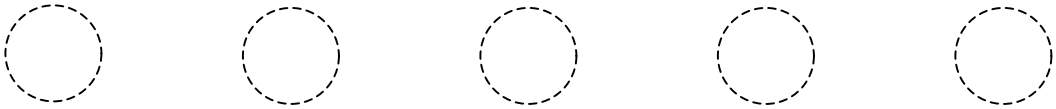
2



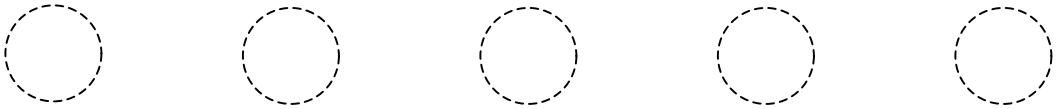
3



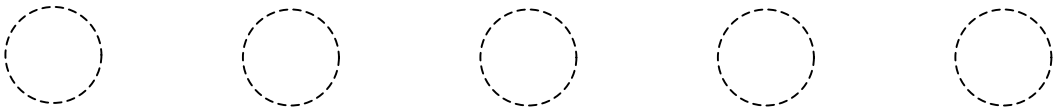
4



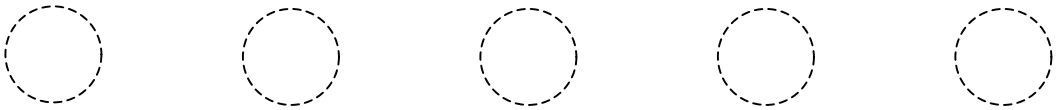
5



7



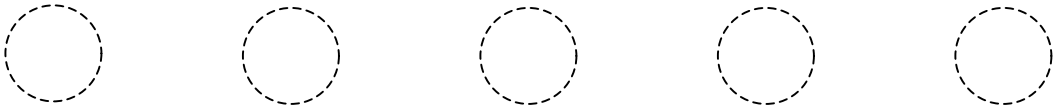
8



9

10

11



12

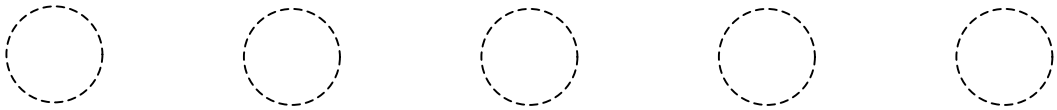
13

14

15

16

17



18

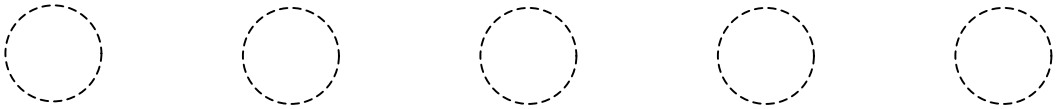
19

20

21

22

23



24