OCR GCSE Computer Science

Your notes

Computational Thinking, Searching & Sorting Algorithms

Contents

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Principles of Computational Thinking

Your notes

- Solving problems that can be implemented by a computer system is known as **computational thinking**
- There are three main principles of computational thinking:
  - **Abstraction**
  - **Decomposition**
  - **Algorithmic thinking**

Abstraction

What is abstraction?

- Abstraction is the process of **removing unnecessary details** of a problem to **focus on the important features** to implement in a solution
- Examples of abstraction include modelling a real-life object, environment, action, sequence of actions or concept.
- Implementations of these include:
  - **a computer game that simulates playing a sport**
  - **a simulator such as a car or flight simulator,**
  - **a map of a bus or train route in a city**
- When creating a program, developers must **identify important features** that will contribute to solving the problem or have a role to play in the solution

Computer games

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Your notes

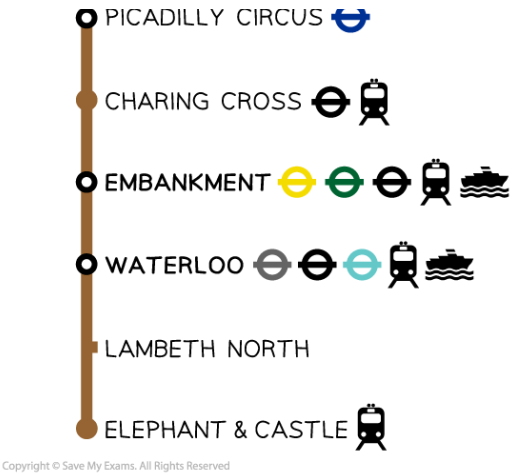
- Computer games use a large amount of abstraction, **removing the elements that a user does not need to consider** in order to enjoy playing the game
- When using abstraction in computer games which are designed to simulate real life, the aim is to make the game realistic and visually appealing whilst keeping the game fun to play
- In a game that simulates a sport, it is important to the user that visually they recognise the environment and when they perform an action, they see a response
- However, users do not need to know the complex algorithms used to control the non player characters (NPCs)

Train map

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Your notes

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers



Copyright © Save My Exams. All Rights Reserved

Your notes

- Another specific example of abstraction would be the London underground train route map; travellers **do not need to know the geographical layout** of the routes, only that getting on at stop A will eventually transport you to stop B

Worked Example

Jack plays rugby at his local rugby club. He wants to create a program to store the results of each rugby match they play and the names of the try scorers.

Define what is meant by abstraction **[2]**

Give one example of how abstraction could be used when developing this program **[1]**

Answer

- Abstraction is removing unnecessary detail from a problem in order to focus on the parts of the problem that need solving
- Simplifies the problem // reduces complexity // easier to solve

Any suitable example of abstraction as long as it is relevant to the system

- Examples of what to ignore/hide/remove:
  - Time the try was scored
  - Player shirt number
  - Venue
- Examples of parts to focus on:
  - Player name
  - Match result
  - Tries scored

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Decomposition

What is decomposition?


Your notes

- Decomposition is the process of **breaking down a large problem** into **a set of smaller problems**
- Benefits of decomposition are:
  - **Smaller problems are easier to solve**
  - **Each smaller problem can be solved independently of the others**
  - **Smaller problems can be tested independently**
  - **Smaller problems can be combined to produce a solution to the full problem**
- An examples of decomposition in computing is:

Computer games

- Modern computer games are decomposed to **break down the complexity** of the problem into more **manageable** '**chunks**'
- Creating an entire game at once would be **challenging** and **inefficient**, so it could be decomposed into:
  - **Levels** - Levels can be designed/created/tested/ independently of other levels
  - **Characters** - The mechanics of characters in the game can be designed and created by a separate team
  - **Landscape** - The art team can work on the visual aspects of the game without needing to understand how the game is programmed
- Once all of the smaller problems are completed, joined together a complex game has been created

Algorithmic Thinking

What is algorithmic thinking?

- Algorithmic thinking is the process of **creating step-by-step instructions in order to produce a solution to a problem**
- Algorithmic thinking requires the use of **abstraction** and **decomposition** to identify each individual step
- Once each step has been identified, **a precise set of rules (algorithm)** can be created and the problem will be solved

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

- An example of algorithmic thinking is following a recipe, if the recipe is followed **precisely** it should lead to the desired outcome
- A set of traffic lights is an example of how algorithmic thinking can lead to **solutions being automated**

Your notes

Examiner Tips and Tricks

Don't just memorise the definition of algorithmic thinking! In the exam, you will have to demonstrate the skill by breaking a problem down into clear, step-by-step instructions (writing algorithms)

Algorithmic thinking takes practice, so the more you do, the easier it will become!

Worked Example

State the name of each of the following computational thinking techniques.

1. Breaking a complex problem down into smaller problems. **[1]**

2. Hiding or removing irrelevant details from a problem to reduce the complexity. **[1]**

Answers

- Decomposition
- Abstraction

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Standard Searching Algorithms

Your notes

Binary Search

What is a searching algorithm?

- Searching algorithms are **precise step-by-step instructions** that a computer can follow to **efficiently locate specific data** in massive datasets
- Two common searching algorithms are:
  - **Binary search**
  - **Linear search**

What is a binary search?

- A binary search keeps **halving a dataset by comparing the target value with the middle value**, going **left if smaller**, **right if bigger**, until it finds the value or realises it's not there
- To perform a binary search the **data must be in order!**
- A binary search can be performed on datasets containing **numbers or words**.
- Searching for a word instead of a number is the same process, except comparisons are made based on position in the alphabet (**alphabetically**) instead of numerical size

How do you perform a binary search?

Step	Instruction
1	Identify the **middle value**
2	**Compare** to the value you are looking for
3	**IF** it is the value you are looking for... ▪ Stop!
4	**ELSE IF** is it bigger than the one you are looking for... ▪ Create a new list with the values on the left of the middle value

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

5	**IF** it is smaller than the one you are looking for... ▪ Create a new list with the values on the right of the middle value
6	**REPEAT** with the new list

Your notes

Example 1 – numbers

- Perform a **binary search** to locate number **7** in the following dataset

2	5	7	12	15	22	46

Step	Instruction
1	Identify the **middle value (12)**
	2 | 5 | 7 | 12 | 15 | 22 | 46
2	**Compare** to the value you are looking for - **Is 12 == 7?**
3	**IF** it is the value you are looking for... ▪ Stop! - **12 is not equal to 7**
4	**ELSE IF** is it bigger than the one you are looking for... **Is 12 > 7? YES** ▪ Create a new list with the values on the left of the middle value 2 | 5 | 7
5	**IF** it is smaller than the one you are looking for... ▪ Create a new list with the values on the right of the middle value
6	**REPEAT** with the new list 2 | 5 | 7

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Your notes

	Is 5 == 7? NO **Is 5 > 7? NO - create new list with values to the right** **7** **Is 7 == 7? YES** **STOP!**

Example 2 - words

- Perform a **binary search** to locate the word **"Rock"** in the following dataset

Ballroom	**Country**	**Electronic**	**Hip Hop**	**Jazz**	**Rock**	**Techno**

Step	Instruction
1	Identify the **middle value ("Hip Hop")**
	Ballroom | **Country** | **Electronic** | **Hip Hop** | **Jazz** | **Rock** | **Techno**
2	**Compare** to the value you are looking for - **Is "Hip Hop" == "Rock"?**
3	**IF** it is the value you are looking for... ▪ Stop! - **"Hip Hop" is not equal to "Rock"**
4	**ELSE IF** is it bigger than the one you are looking for... **Is "Hip Hop" > "Rock"? NO** ▪ Create a new list with the values on the left of the middle value
5	**IF** it is smaller than the one you are looking for... **Is "Hip Hop" < "Rock"? YES** ▪ Create a new list with the values on the right of the middle value

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

	Jazz	Rock	Techno
6	**REPEAT** with the new list Jazz | Rock | Techno **Is "Rock" == "Rock"? YES** **STOP!**		

Your notes

Examiner Tips and Tricks

If the dataset has an even number of values, the simplest way to identify the middle is to divide the total values by 2 and use that as a middle value i.e. a dataset with 8 values, 4 would be the middle value

Worked Example

Describe the steps a binary search will follow to look for a number in a sorted list **[4]**

Answer

- Select / choose / pick **middle** number (or left/right of middle as even number) and ...
- ...check if selected number is **equal to / matches** target number (not just compare)
- ...if searched number is **larger**, discard left half // if searched number is **smaller**, discard right half
- Repeat until **number found**
- ... or remaining list is of size 1 / 0 (number not found)

Guidance

- Can get a mark for bullet points 1 & 2 in one step (e.g. check if the middle value is the one we're looking for")

A binary search in python

```
# Identify the dataset to search, the target value and set the initial flag
data = [2, 4, 6, 8, 10, 12, 14]
```


© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

```
target = 8
found = False

# Set the initial low and high pointers to the beginning and end of the data
low = 0
high = len(data) - 1

# While the low pointer is less than or equal to the high pointer
while found is False and low <= high:

  # Find the middle index
  mid = (low + high) // 2

  # Check if the target is at the middle index
  if data[mid] == target:

    # If the target is found, output a message
    found = True
    print("Target found")

  # If the target is less than the middle value, search in the left half of the data
  elif data[mid] > target:
    high = mid - 1

  # Otherwise, search in the right half of the data
  else:
    low = mid + 1

# If the target is not found, output a message
if found is False:
  print("Target not found")
```


Your notes

Linear Search

What is a linear search?

- A linear search **starts with the first value** in a dataset and **checks every value one at a time** until all values have been checked
- A linear search can be performed **even if the values are not in order**

How do you perform a linear search?

Step	Instruction

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

1	Check the first value
2	**IF** it is the value you are looking for ▪ STOP!
3	**ELSE** move to the next value and check
4	**REPEAT UNTIL** you have checked all values and not found the value you are looking for


Your notes

Examiner Tips and Tricks

You will not be asked to perform a linear search on a dataset in the exam, you will be expected to understand how to do it and know the advantages and disadvantages compared to a binary search

What are the advantages and disadvantages of searching algorithms?

Searching Algorithm	Advantages	Disadvantages
Binary search	▪ **Fast** for large datasets ▪ **Efficient** for repeated searches	▪ Dataset must be in **order** ▪ More **complex** to implement
Linear search	▪ Works on **unsorted** datasets ▪ **Faster** (than binary) on **very small** datasets ▪ **Simple** to understand and implement	▪ **Slow** for large datasets ▪ **Inefficient**, starts at the beginning each time

Worked Example

A linear search could be used instead of a binary search.

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Describe the steps a linear search would follow when searching for a number that is **not** in the given list **[2]**

Answer

- Starting with the **first value**
- Checking **all values in order**

Guidance

- Must cover idea of checking all value AND being done in order!
- "Checks each value from the beginning to the end" implies order so would get both bullet point 1 & 2


Your notes

A linear search in python

```
# Identify the dataset to search, the target value and set the initial flag
data = [5, 2, 8, 1, 9]
target = 11
found = False

# Loop through each element in the data
for index in range(0,len(data) - 1):

  # Check if the current element matches the target
  if data[index] == target:

    # If found, output message
    found = True
    print("Target found")

#If the target is not found, output a message
if found is False:
  print("Target not found")
```


© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Standard Sorting Algorithms


Your notes

What is a sorting algorithm?

- In GCSE Computer Science, sorting algorithms are **precise step-by-step instructions** that a computer can follow to sort data in massive datasets efficiently
- Three common sorting algorithms are:
  - **Bubble sort**
  - **Merge sort**
  - **Insertion sort**

Bubble Sort

What is a bubble sort?

- A bubble sort is a simple sorting algorithm that starts at the beginning of a dataset and **checks values** in **'pairs'** and **swaps them** if they are **not in the correct order**
- One full run of comparisons from beginning to end is called a '**pass**', a bubble sort may require multiple '**passes**' to sort the dataset
- The algorithm is finished when there are **no more swaps to make**

How do you perform a bubble sort?

Step	Instructions
1	Compare the first two values in the dataset
2	**IF** they are in the wrong order... ▪ Swap them
3	Compare the next two values
4	**REPEAT** step 2 & 3 until you reach the end of the dataset (**pass 1**)

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

5	**IF** you have made any swaps... ▪ **REPEAT** from the start (pass 2,3,4...)
6	**ELSE** you have not made any swaps... ▪ STOP! the list is in the correct order


Your notes

Example

- Perform a bubble sort on the following dataset

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

5	2	4	1	6	3

Step	Instruction
1	Compare the first two values in the dataset 5 2 4 1 6 3
2	**IF** they are in the wrong order... • Swap them 2 5 4 1 6 3
3	Compare the next two values 2 5 4 1 6 3
4	**REPEAT** step 2 & 3 until you reach the end of the dataset • 5 & 4 SWAP! 2 4 5 1 6 3 • 5 & 1 SWAP! 2 4 1 5 6 3 • 5 & 6 NO SWAP! 2 4 1 5 6 3 • 6 & 3 SWAP! 2 4 1 5 3 6 • End of pass 1

Your notes

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

<table>
<tr><td>5</td><td>IF you have made any swaps...
• REPEAT from the start
• End of pass 2 (swaps made)
<table><tr><td>2</td><td>1</td><td>4</td><td>3</td><td>5</td><td>6</td></tr></table>• End of pass 3 (swaps made)
<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table>• End of pass 4 (no swaps)
<table><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table></td></tr>
<tr><td>6</td><td>ELSE you have not made any swaps...
• STOP! the list is in the correct order</td></tr>
</table>

Your notes

Examiner Tips and Tricks

In the exam you do not have to show every swap that takes place in a bubble sort. You can show the outcome of a bubble sort at the end of each pass. If you have the outcome of each pass correct then a bubble sort has been implemented correctly and all marks will be given!

Worked Example

A program uses a file to store a list of words.

A sample of this data is shown

Milk	Eggs	Bananas	Cheese	Potatoes	Grapes

Show the stages of a bubble sort when applied to data shown **[2]**

How to answer this question

- We need to sort the values in to **alphabetical order** from A-Z
- You **CAN** use the first letter of each word to simplify the process

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Answer

- E, B, C, M, G, P (pass 1)
- B, C, E, G, M, P (pass 2)

Your notes

A bubble sort in python

```
# unsorted dataset
nums=[66, 7, 69, 50, 42, 80, 71, 321, 67, 8, 39]

# count the length of the dataset
numlength = len(nums)

# sets a flag to initiate the loop
swaps = True

while swaps == True :
   swaps = False
   # loop through the dataset
   for y in range(numlength-1) :
      # if the first number is bigger than the second number
      if nums[y] > nums[y+1] :
         # swap the numbers using a temporary variable
         temp = nums[y]
         nums[y] = nums[y+1]
         nums[y+1] = temp
         # sets the flag to true
         swaps = True

# prints the sorted list
print (nums)
```

Merge Sort

What is a merge sort?

- A merge sort is a sorting algorithm that uses the '**divide and conquer**' strategy of **dividing a dataset** into smaller **sub-datasets** and **merging** them back together in the correct order

How do you perform a merge sort?

Step	Instruction
1	**Divide** the dataset into individual datasets by repeatedly splitting the dataset in half (**DIVIDE**)
2	**Merge** pairs of sub-datasets together by comparing the first value in each dataset (**CONQUER**)

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

3	**REPEAT** step 2 until all sub-datasets are merged together into one dataset

Your notes

Example

- Perform a merge sort on the following dataset

7	4	1	2	6	3	8	5

Step	Instruction
1	**Divide** the dataset into individual datasets by repeatedly splitting the dataset in half (**DIVIDE**) 7 4 1 2 6 3 8 5 divide in half (**2 datasets of 4 values**) 7 4 1 2 | 6 3 8 5 divide in half again (**4 datasets of 2 values**) 7 4 | 1 2 | 6 3 | 8 5 divide in half again (**8 datasets of 1 value**) 7 | 4 | 1 | 2 | 6 | 3 | 8 | 5
2	**Merge** pairs of sub-datasets together by comparing the first value in each dataset (**CONQUER**) 7 4 | 1 2 | 6 3 | 8 5 Merge into 4 datasets of 2 values 4 7 | 1 2 | 3 6 | 5 8
3	**REPEAT** step 2 until all sub-datasets are merged together into one dataset Merge into 2 datasets of 4 values 1 2 4 7 | 3 5 6 8 Merge into 1 dataset of 8 values (in the correct order) 1 2 3 4 5 6 7 8

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

Examiner Tips and Tricks

In the exam, the divide stage could already be done for you and you would only need to demonstrate the conquer stage!

Your notes

Insertion Sort

What is an insertion sort?

- The insertion sort **sorts one item at a time** by **placing it in the correct position** of an unsorted list. This process repeats until all items are in the correct position
- Values in the dataset can move **as many places as they need**

How do you perform an insertion sort?

Step	Instruction
1	Take the **first value** in the dataset, this is now the **sorted list**
2	Look at the **next value** in the dataset and **compare it to the first value** **IF** it is smaller ▪ insert it into the correct position to the left
3	**ELSE** ▪ Keep in the same position
4	**REPEAT** steps 2 & 3 until all values in the dataset have been compared, the list should now be in order

Example

- Perform an insertion sort on the following dataset

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

54	27	17	9	40	12

Step	Instruction
1	Take the **first value** in the dataset, this is now the **sorted list** 54 | 27 | 17 | 9 | 40 | 12
2	Look at the **next value** in the dataset (**27**) and **compare it to the first value** (is 27 < 54?) **IF** it is smaller (**YES!**) • insert it into the correct position on the left of the first 27 | 54 | 17 | 9 | 40 | 12
3	**ELSE** • Keep in the same position
4	**REPEAT** steps 2 & 3 until all values in the dataset have been compared, the list should now be in order 27 | 54 | 17 | 9 | 40 | 12 17 | 27 | 54 | 9 | 40 | 12 9 | 17 | 27 | 54 | 40 | 12 9 | 17 | 27 | 40 | 54 | 12 9 | 12 | 17 | 27 | 40 | 54

Your notes

Examiner Tips and Tricks

Remember that in any sorting algorithm question, you make be asked to demonstrate a sort using words instead of numbers. The process is the same, the order will be alphabetically from A-Z unless stated in the question.

© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers

> You may also abbreviate words using just their first letter as long as two values don't share the same first letter!

Your notes

An insertion sort in Python

```
# Initialise the list to be sorted
numbers = [12, 11, 13, 5, 6]

# Get the length of the list
length = len(numbers)

# Perform insertion sort
for i in range(1, length):
    key = numbers[i]  # The current element to be inserted in the sorted part of the array
    sorted_index = i - 1  # The last index of the sorted part of the array

    # Move elements of the sorted part of the array that are greater than the key
    # to one position ahead of their current position
    while sorted_index >= 0 and numbers[sorted_index] > key:
        numbers[sorted_index + 1] = numbers[sorted_index]
        sorted_index = sorted_index - 1

    # Place the key in its correct position in the sorted part of the array
    numbers[sorted_index + 1] = key

# Print the sorted list
print("Sorted list:", numbers)
```


© 2015–2025 Save My Exams, Ltd. · Revision Notes, Topic Questions, Past Papers