



# Dynamic data structure: advantage

Which of these is an **advantage** of a dynamic data structure?

- The memory allocation is fixed so there is no need to check size when adding and removing data items
  - It allocates memory to pointers to keep track of where things are
  - It makes efficient use of memory, using only as much memory as it needs
  - It will overflow if it exceeds its allocated memory
- 
- 
- 

---

---

---

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Dynamic vs static structures 2

Data structures can be static or dynamic. Which three of the statements are **correct**?

- A static array cannot be resized at runtime.
  - A static array uses an allocation of contiguous (one after another) locations in main memory.
  - A dynamic data structure is a collection of data in memory that can grow or shrink in size at runtime.
  - The elements of a dynamic linked list can be accessed directly by index.
- 
- 
- 

Quiz:

[STEM SMART Computer Science Week 6](#)

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Array: 2D indexing

The table below illustrates the contents of a two-dimensional array that has been set up to store words for an application to help schoolchildren to learn their spellings.

Index	0	1	2	3
0	school	pull	where	here
1	path	floor	sugar	bread
2	accident	answer	eight	critical

When implemented in the program, the name of the array is `spelling_words`. The first index references the row number and the second index references the column number.

What will be displayed when the following line of code (shown in pseudocode) is run?

`PRINT(spelling_words[2,1])`

- answer
  - sugar
  - path
  - pull
- 
- 
- 

Quiz:

[STEM SMART Computer Science Week 6](#)

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Array: trace pseudocode 1

Emily is organising a charity event at her school. She needs to select four students to meet representatives from the charity when they come to visit.

She starts by creating an array **students** to hold the names of the eight students who raised money for the charity:

After she has added all the names, the array (in pseudocode) looks like this:

```
["Miguel", "Laura", "Mariam", "Arthur", "Musa", "Magda", "Ben", "Diane"]
```

Now Emily writes a program (expressed in pseudocode below) to pick four students:

## Pseudocode

```
1 PROCEDURE pick_students()
2     students = ["Miguel", "Laura", "Mariam", "Arthur", "Musa", "Magda", "Ben", "Diane"]
3     selected = 0
4
5     WHILE selected < 4
6         picked = RANDOM_INT(0,7) // Return a random integer between 0 and 7
7         IF students[picked] != "Selected" THEN
8             PRINT(students[picked])
9             students[picked] = "Selected"
10            selected = selected + 1
11        ENDIF
12    ENDWHILE
13 ENDPROCEDURE
```

When the program is run, the sequence of random numbers is:

4, 1, 7, 1, 3

What is the output of the program?

- Arthur Miguel Ben Mariam
- Musa Laura Diane Selected Arthur
- Musa Laura Diane Laura Arthur
- Musa Laura Diane Arthur



# Dictionary: definition 2

In a dictionary, items are accessed by  whereas items stored in an array are accessed by . A dictionary is sometimes called an  array.

Arrays are useful when you want to  over all the stored values because they are stored  in main memory. Dictionaries are a better choice if you want to access discrete values because you retrieve them by  rather than by .

Items:

---

---

---

Quiz:

[STEM SMART Computer Science Week 6](#)

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Dictionaries: compared to arrays

What is the primary difference between an array and a dictionary in computer science?

- Arrays store data in key/value pairs, while dictionaries store data as a list of elements.
  - Arrays use numeric indices whereas data in a dictionary is retrieved by key.
  - Arrays can only store primitive data types, while dictionaries can store objects and structures.
- 
- 
- 

Quiz:

[STEM SMART Computer Science Week 6](#)

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Record: access 1

You are creating a system for a local charity and will use a record structure to store the information for each volunteer.

The definition of a suitable record structure to store the volunteer data is shown below.

## Pseudocode

```
1 VolunteerRecord = RECORD
2     volunteer_id: Integer
3     first_name: String
4     last_name: String
5     date_of_birth: Date
6     email: String
7     active: Boolean
8 ENDRECORD
```

The following pseudocode shows the process of storing the details of a volunteer called Raj Whelan.

## Pseudocode

```
1 volunteer1 = VolunteerRecord
2
3 volunteer1.volunteer_id = 1
4 volunteer1.first_name = "Raj"
5 volunteer1.last_name = "Whelan"
6 volunteer1.date_of_birth = "05/07/1985"
7 volunteer1.email = "RajWhelan@abc.com"
8 volunteer1.active = True
```

A username is generated automatically for each volunteer so that they can log in to the charity's system. The format of the username is **the last name followed by the first name without any spaces**.

Which one of the following choices would generate the correct username for Raj Whelan?

- `last_name + first_name`
- `volunteer1.last_name + volunteer1.first_name`

- `first_name + last_name`
  - `volunteer1.first_name + volunteer1.last_name`
- 
- 
- 

Quiz:

**STEM SMART Computer Science Week 6**

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





# Record: array of records – advantages

Rudi has made a list of gifts that he would like to buy for his family to celebrate the festive break. He has written a program to help him keep track of his ideas. For each gift idea he wants to store:

- Name of the person (the gift is for)
- Description of the gift
- Likely cost
- Purchased (yes/no)

He has decided that an **array of records** is a suitable data structure for his program.

Why has Rudi decided that a record structure is a suitable container for each gift idea?

- There is no limit to the number of records he can store.
  - It can hold data of different data types.
  - The elements of the record (the fields) can be referred to using an index number.
- 
- 
- 

Quiz:

[STEM SMART Computer Science Week 6](#)

---

All teaching materials on this site are available under a CC BY-NC-SA 4.0 license, except where otherwise stated.





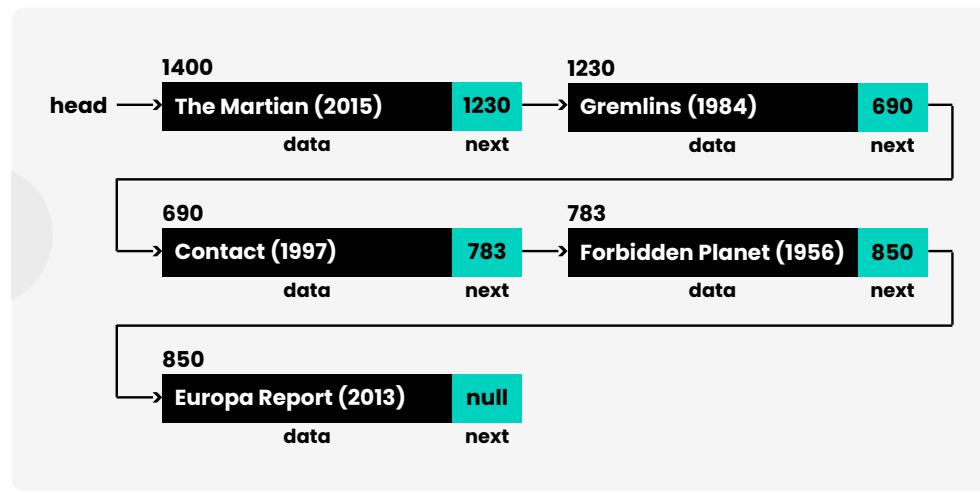
# Linked list: add a node to an unordered list

Matthew has designed a system that keeps a list of films he wants to watch. When he hears about an interesting new film, he adds it to the list. When he has watched a film, he deletes it from the list.

The list of films is stored within the system as a linked list:

- Each node has a distinct memory location, which is shown in **Figure 1** as the number above the node
- Each node contains data (in this case, the name of the film) and a pointer to the memory location of the next node
- The node labelled **head** indicates the first element in the linked list (the **head** of the list)
- For the last element in the list, the next node pointer always points to a **null** value to mark the end of the list
- The list is **unordered** so when a new node is added, it is added to the **head** of the list

The current state of the linked list is shown in **Figure 1** below:



**Figure 1:** The current state of the linked list

Matthew's friend recommends a new film called "Spiderman: Across the Spider-verse (2023)", and Matthew uses the system to add it to his list of films to watch.

A new node is created at memory location 250 to store the details of the film. In this new node, the value of **data** will be set to "Spiderman: Across the Spider-verse (2023)". What value will **next** be set to?

Quiz:

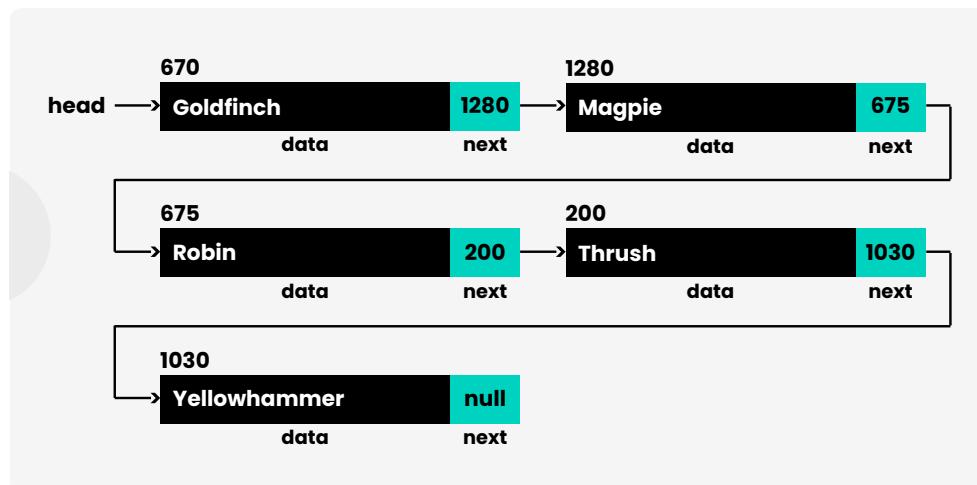


# Linked list: add a node to an ordered list

Eli uses a system that keeps a list of the different birds he sees in his garden. The list of birds is stored within the system as a linked list:

- Each node has a distinct memory location, which is shown in **Figure 1** as the number above the node.
- Each node contains data (in this case, the name of the bird) and a pointer to the memory location of the next node.
- The node labelled **head** indicates the first element in the linked list (the **head** of the list).
- For the last element of the list, the next node pointer always points to a **null** value to mark the end of the list.
- The list is **ordered alphabetically (A to Z)**. When a new node is added, it is added **into the correct position** within the list.

The current state of the linked list is shown in **Figure 1** below:



**Figure 1:** The current state of the linked list

Eli spots a pigeon in the garden and records it in the system.

A new node is created at memory location 950 to store the details of the bird. In this new node, the value of **data** will be set to "Pigeon". What value will **next** be set to?