

Data Types and Expressions

Data

Data

Expressions

Data

Expressions

Control Flow

Data

- int
- float
- double
- char
- string
- bool
- vector
- Programmer defined classes

Expressions

- I/O expressions
- Arithmetic expressions
- Boolean expressions

Control Flow

- Sequential
- Branching
 - if
 - if-else
 - if-else if-else
 - switch
- Iterative
 - while
 - For
- Function calls
- Exceptions

Data

Expressions

Control Flow

```
/* This program reads two integers from the user and  
prints their sum */
```

```
#include <iostream>
```

```
using namespace std;
```

```
int main() {
```

```
    int num1;                //will hold the first input
```

```
    int num2;                //will hold the second input
```

```
    int sum;                 //will hold the sum
```

```
    cout<<"Please enter two numbers separated by a space"<<endl;
```

```
    cin>>num1>>num2;
```

```
    sum = num1 + num2;
```

```
    cout<<num1<<" + "<<num2<<" = "<<sum<<endl;
```

```
    return 0;
```

```
}
```

Data

Expressions

Control Flow

Data

Expressions

Control Flow

- Sequential

Data

Expressions

- I/O expressions

Control Flow

- Sequential

Data

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

Data

- int

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

Data

- int
- float
- double
- char
- string
- bool

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

Data

- int

Expressions

- I/O expressions
- Arithmetic expressions

Control Flow

- Sequential

The `int` Data Type

Integer data type

Whole numbers, positive and negative

Can be used for counting

Can be used for indexing

Can be used for arithmetic

Can be used for logical operations

Can be used for bitwise operations

Can be used for memory management

The `int` Data Type

Kind of data:

The `int` Data Type

Kind of data: Integer numbers

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

The `int` Data Type

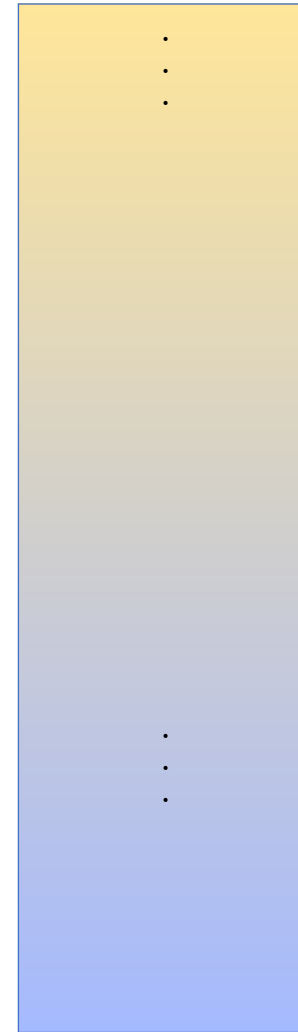
Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)

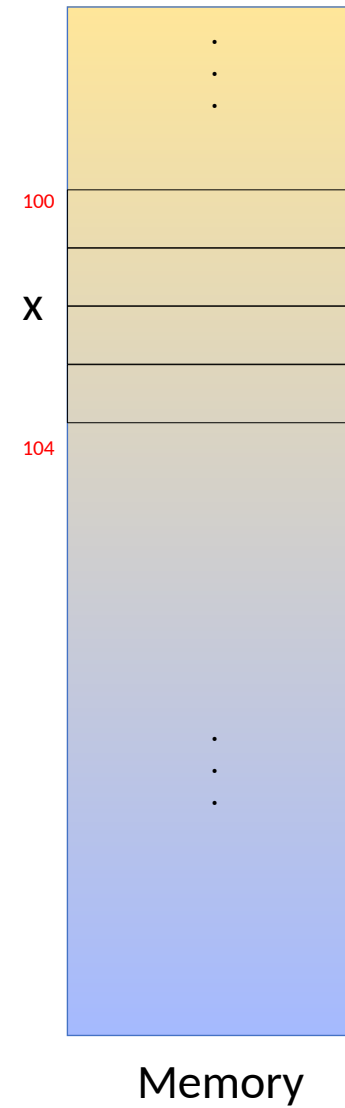
```
int main() {  
    int x;  
  
    return 0;  
}
```

```
int main() {  
    int x;  
  
    return 0;  
}
```

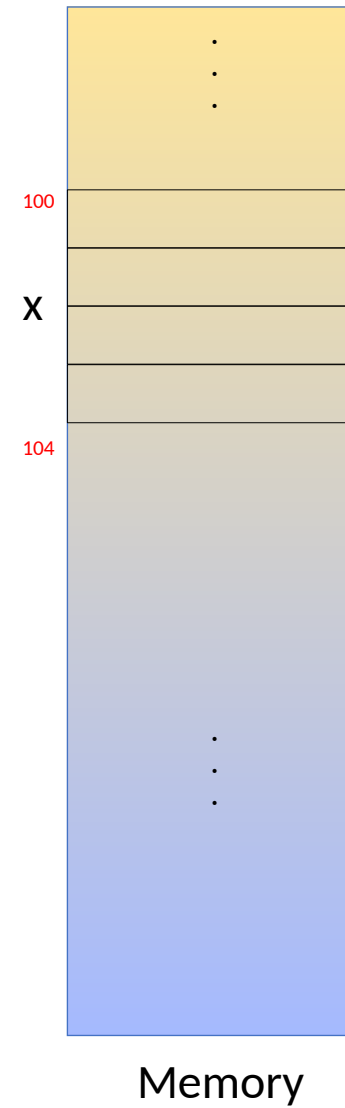


Memory

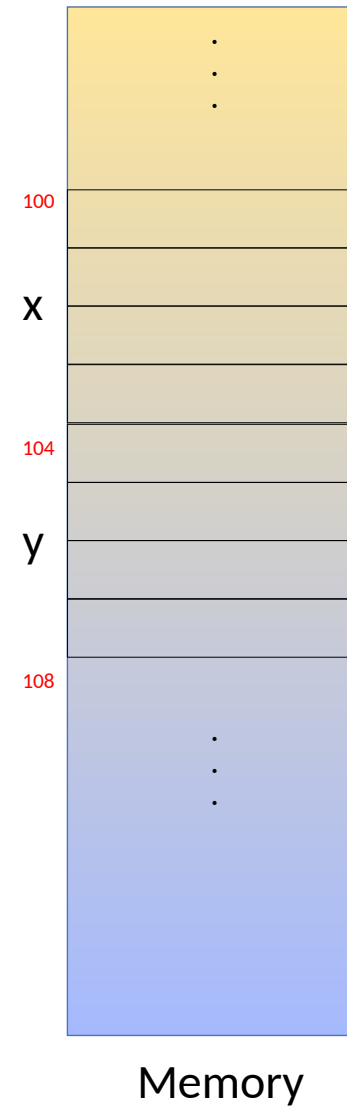
```
int main() {  
    int x;  
  
    return 0;  
}
```



```
int main() {  
    int x;  
    int y;  
  
    return 0;  
}
```



```
int main() {  
    int x;  
    int y;  
  
    return 0;  
}
```



The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)

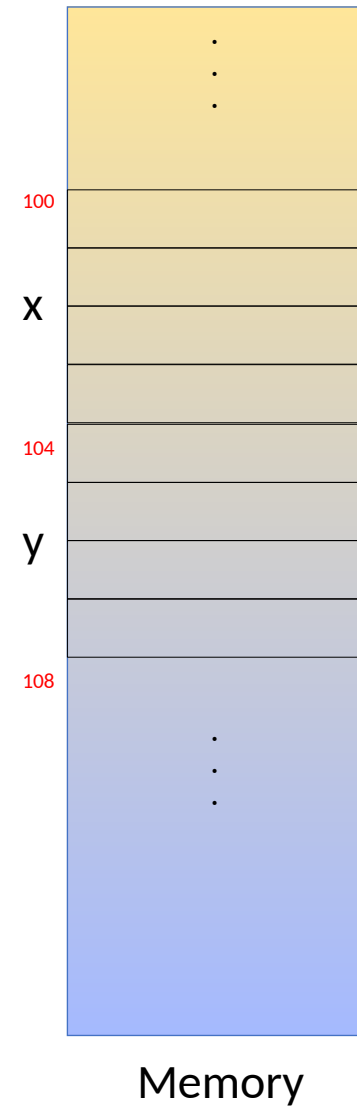
The `int` Data Type

Kind of data: Integer numbers

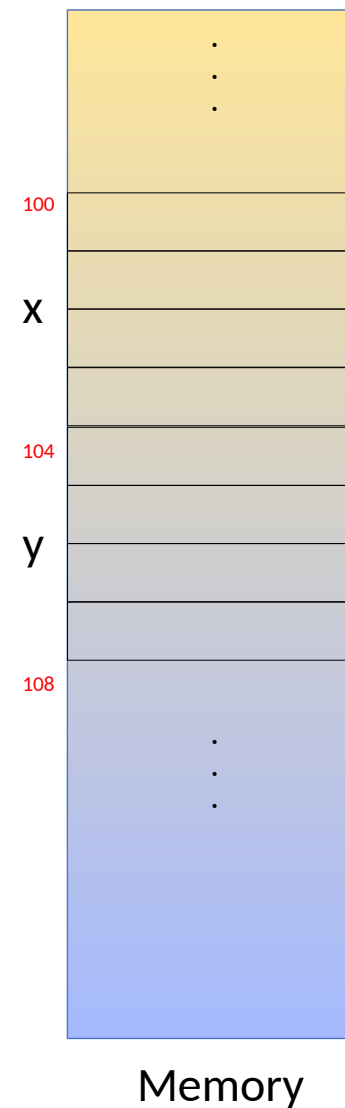
Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

```
int main() {  
    int x;  
    int y;  
  
    return 0;  
}
```

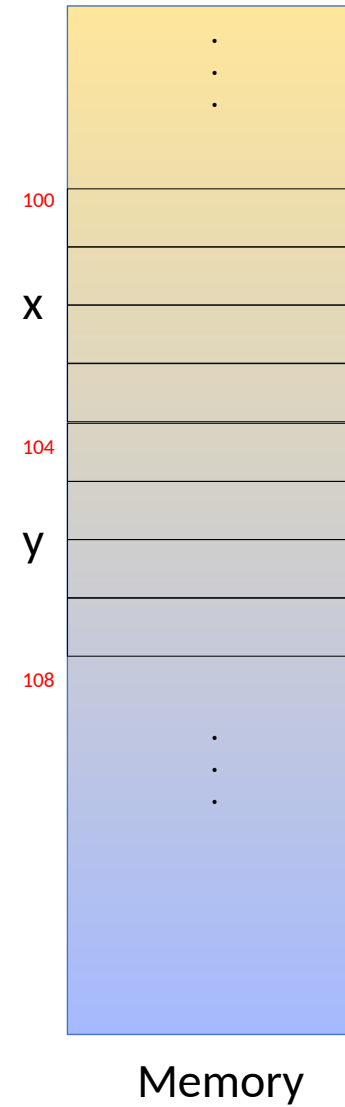


```
int main() {  
    int x;  
    int y;  
  
    x = 6;  
  
    return 0;  
}
```



```
int main() {  
    int x;  
    int y;  
  
    x = 6;  
  
    return 0;  
}
```

$$(6)_{10} = (110)_2$$



```

int main() {
    int x;
    int y;

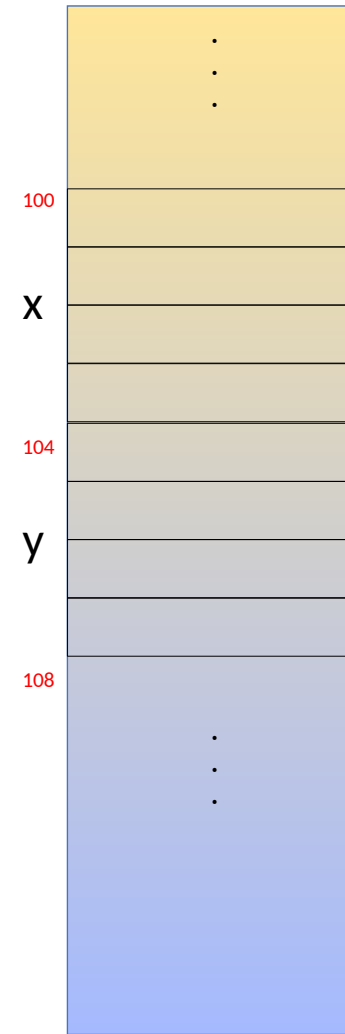
    x = 6;

    return 0;
}

```

$$(6)_{10} = (110)_2$$

$$(6)_{10} = (00000000\ 00000000\ 00000000\ 00000110)_{2's}$$



Memory

```

int main() {
    int x;
    int y;

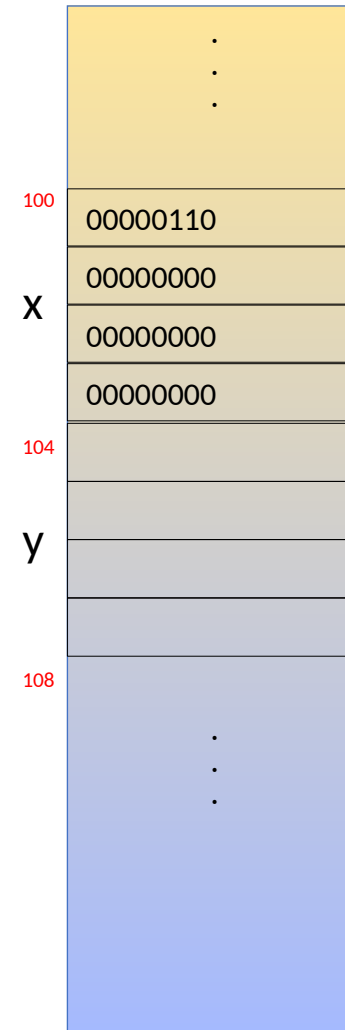
    x = 6;

    return 0;
}

```

$$(6)_{10} = (110)_2$$

$$(6)_{10} = (00000000\ 00000000\ 00000000\ 00000110)_{2's}$$



Memory

```

int main() {
    int x;
    int y;

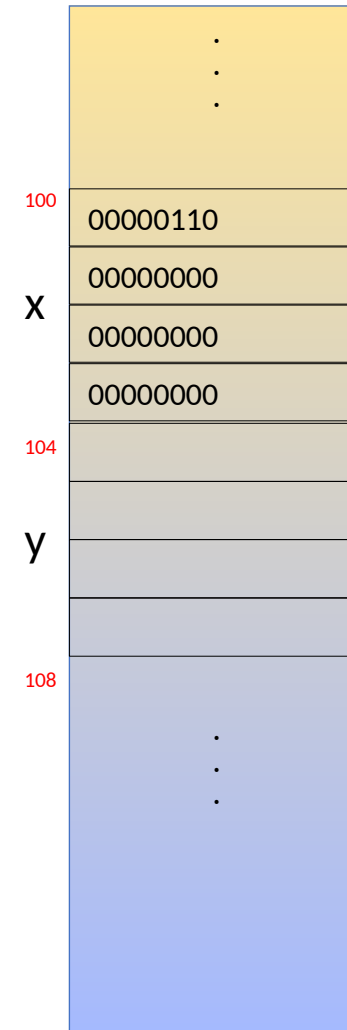
    x = 6;
    y = -6;

    return 0;
}

```

$$(6)_{10} = (110)_2$$

$$(6)_{10} = (00000000\ 00000000\ 00000000\ 00000110)_{2's}$$



Memory


```

int main() {
    int x;
    int y;

    x = 6;
    y = -6;

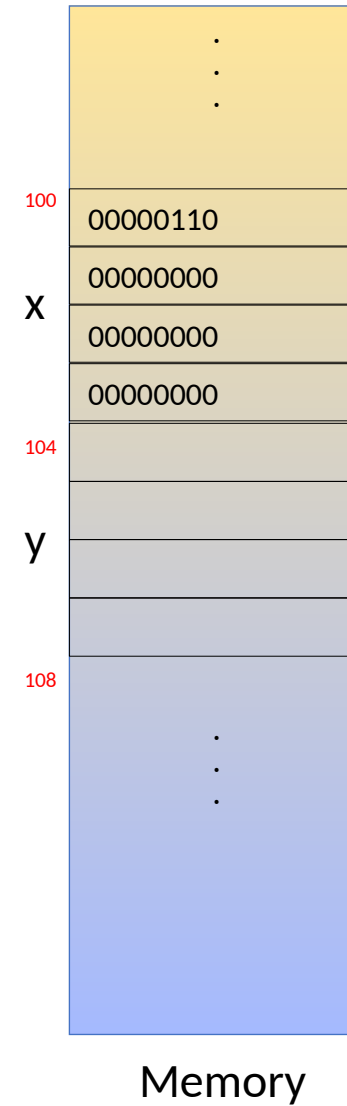
    return 0;
}

```

$$(6)_{10} = (110)_2$$

$$(6)_{10} = (00000000\ 00000000\ 00000000\ 00000110)_{2's}$$

$$(-6)_{10} = (11111111\ 11111111\ 11111111\ 11111010)_{2's}$$



```

int main() {
    int x;
    int y;

    x = 6;
    y = -6;

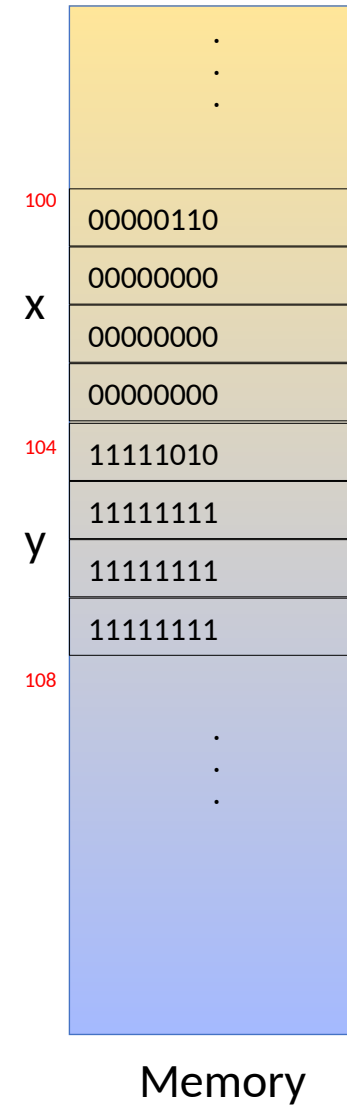
    return 0;
}

```

$$(6)_{10} = (110)_2$$

$$(6)_{10} = (00000000\ 00000000\ 00000000\ 00000110)_{2's}$$

$$(-6)_{10} = (11111111\ 11111111\ 11111111\ 11111010)_{2's}$$



The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals:

Forms of Data

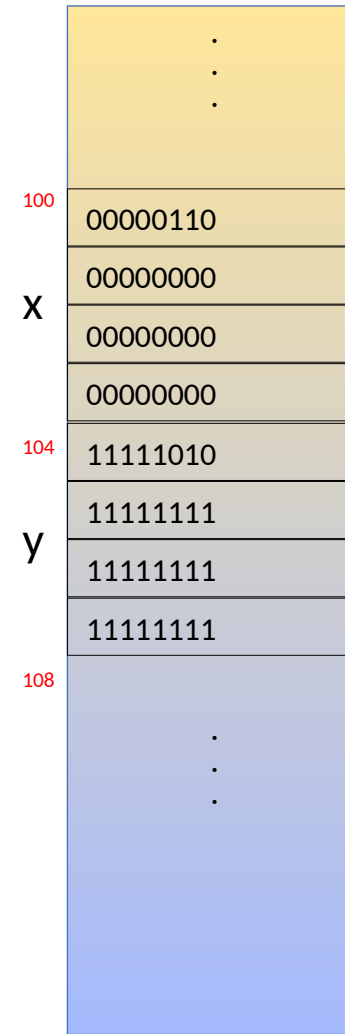
Forms of Data

```
graph TD; A[Forms of Data] --> B[Constants]; A --> C[Variables];
```

Constants

Variables

```
int main() {  
    int x;  
    int y;  
  
    x = 6;  
    y = -6;  
  
    return 0;  
}
```



Memory

Forms of Data

```
graph TD; A[Forms of Data] --> B[Constants]; A --> C[Variables];
```

Constants

Variables

```
int x;  
double y;  
...
```

Forms of Data

```
graph TD; A[Forms of Data] --> B[Constants]; A --> C[Variables]; B --> D[C++ Literals];
```

Constants

Variables

```
int x;  
double y;  
...
```

C++ Literals

Forms of Data

```
graph TD; A[Forms of Data] --> B[Constants]; A --> C[Variables]; B --> D[C++ Literals];
```

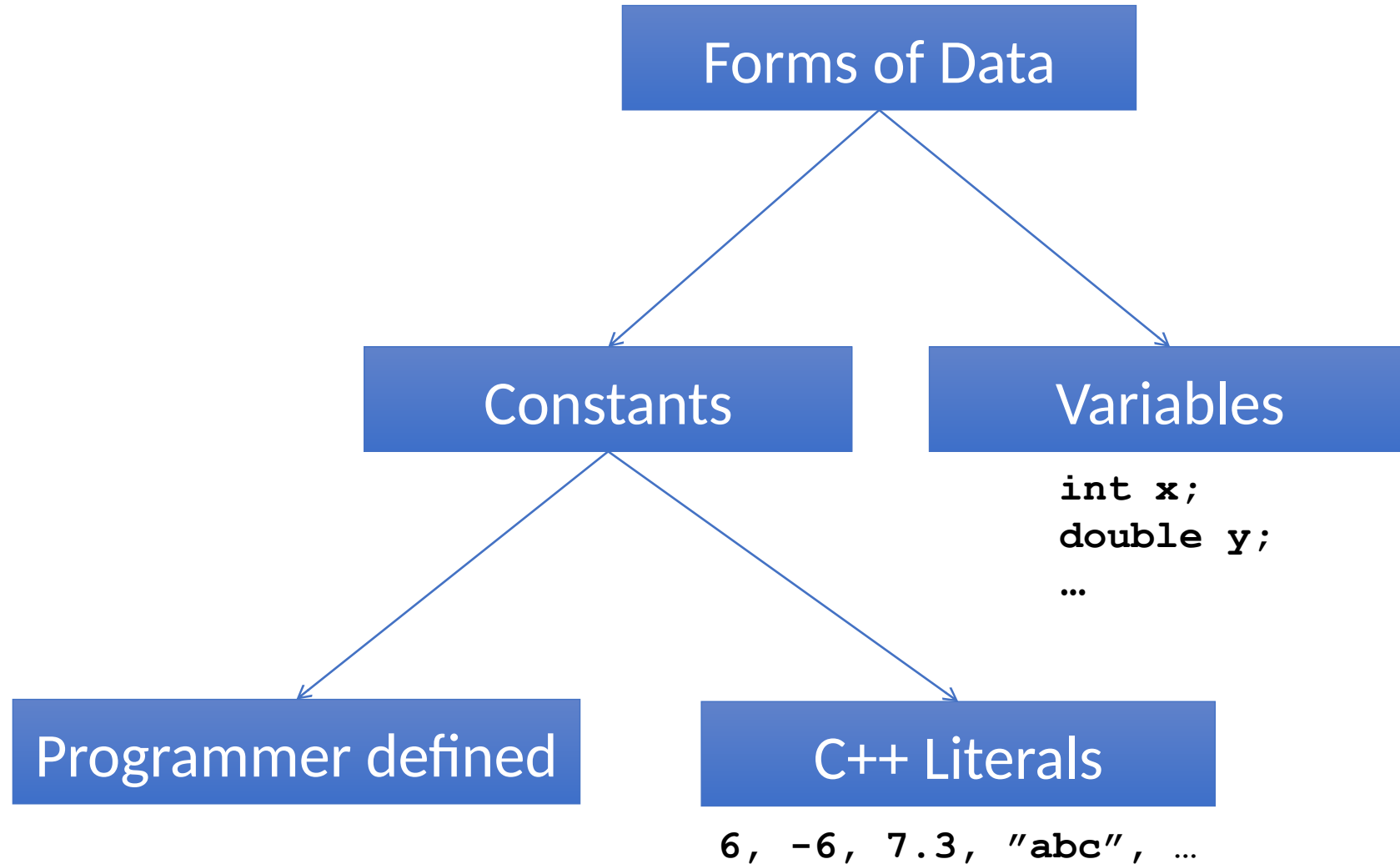
Constants

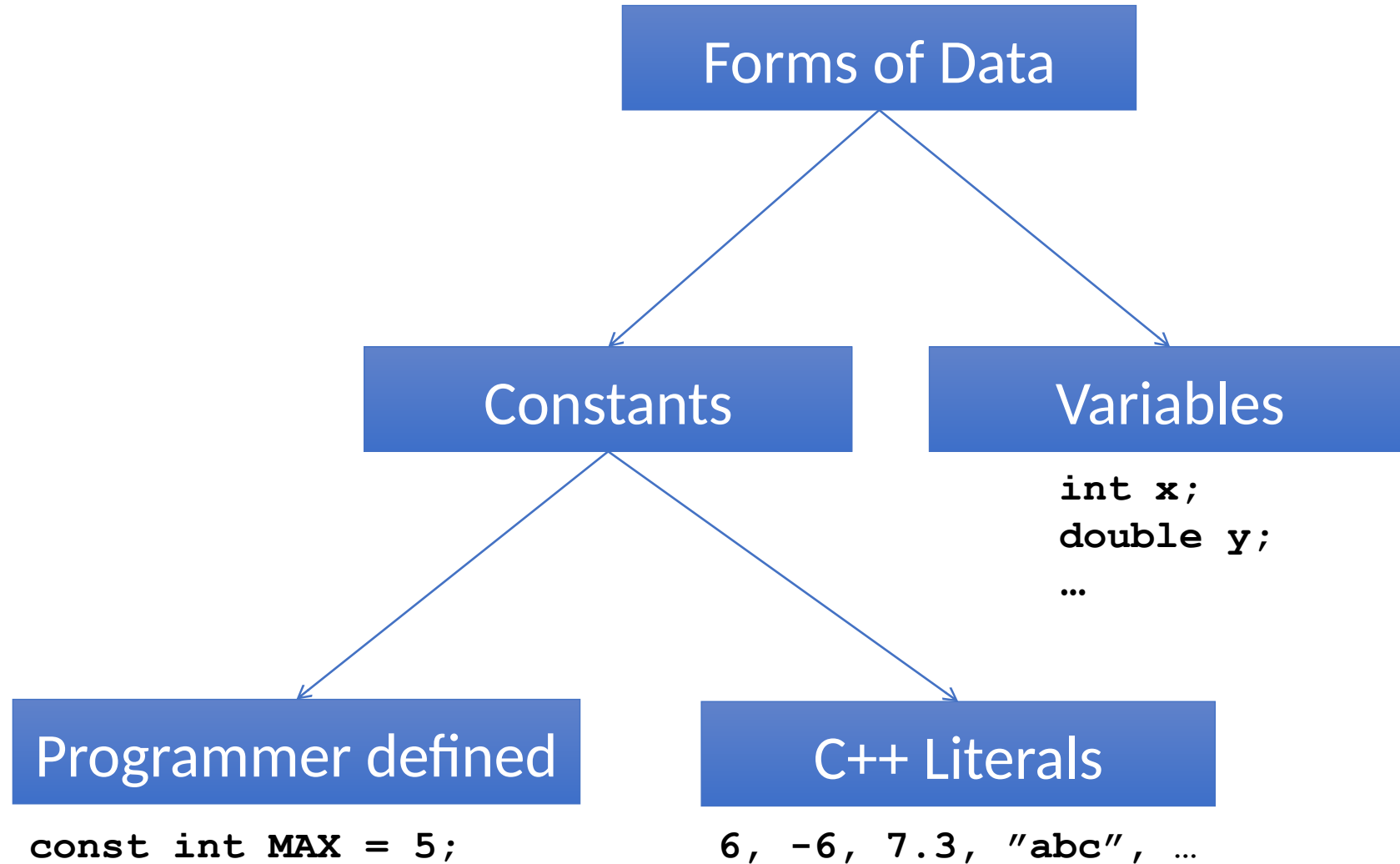
Variables

```
int x;  
double y;  
...
```

C++ Literals

```
6, -6, 7.3, "abc", ...
```





The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals:

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators:

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    return 0;  
}
```


The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    return 0;  
}
```

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *, /

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
  
    return 0;  
}
```

$$13 \div 5$$

$$13 \div 5 = 2R3$$

$$13 \div 5 = 2R3$$

$$13 \text{ div } 5 = 2$$

$$13 \div 5 = 2R3$$

$$13 \text{ div } 5 = 2$$

$$13 \text{ mod } 5 = 3$$

$$13 \div 5 = 2R3$$

In C++

$$13 \text{ div } 5 = 2$$

$$13 \text{ mod } 5 = 3$$

$$13 \div 5 = 2R3$$

In C++

$$13 \text{ div } 5 = 2$$

$$13 / 5$$

$$13 \text{ mod } 5 = 3$$

$$13 \div 5 = 2R3$$

In C++

$$13 \text{ div } 5 = 2$$

$$13 / 5$$

$$13 \text{ mod } 5 = 3$$

$$13 \% 5$$

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
  
    return 0;  
}
```

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
    cout<<x % 2;  
  
    return 0;  
}
```

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *, /, %

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *, /, %, =

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
    cout<<x % 2;  
  
    x = 6;  
  
    return 0;  
}
```

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
    cout<<x % 2;  
  
    x = 6;  
    cout<<x = 7;  
  
    return 0;  
}
```

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
    cout<<x % 2;  
  
    x = 6;  
    cout<<x = 7;  
    y = (x = 8) ;  
  
    return 0;  
}
```

```
int main() {  
    int x;  
    int y;  
  
    x = 5;  
  
    cout<<x + 2;  
    y = x + 2;  
    x + 2;  
  
    cout<<x - 2;  
    y = x * 2;  
  
    cout<<x / 2;  
    cout<<x % 2;  
  
    x = 6;  
    cout<<x = 7;  
    y = (x = 8);  
    y = x = 9;  
    return 0;  
}
```

The `int` Data Type

Kind of data: Integer numbers

Inner representation:

- Each `int` data uses 4 bytes (32 bits)
- The numbers are represented using the 2's complement method

C++ literals: 3, 4, -6, 3954, ...

Arithmetic Operators: +, -, *, /, %, =, ...

Weeks and Days

Write a program that reads from the user the number of days they traveled.

The program will then print their traveling time in the format of full weeks and additional days.

Weeks and Days

Write a program that reads from the user the number of days they traveled.

The program will then print their traveling time in the format of full weeks and additional days.

Example

Please enter number of days you traveled:

Weeks and Days

Write a program that reads from the user the number of days they traveled.

The program will then print their traveling time in the format of full weeks and additional days.

Example

Please enter number of days you traveled:

19

Weeks and Days

Write a program that reads from the user the number of days they traveled.

The program will then print their traveling time in the format of full weeks and additional days.

Example

Please enter number of days you traveled:

19

19 days are 2 weeks and 5 days

Weeks and Days

Weeks and Days

$$19 \div 7$$

Weeks and Days

$$19 \div 7 = 2 \text{ R } 5$$

Weeks and Days

$$19 \div 7 = 2 \text{ R } 5$$

Full weeks



Weeks and Days

$$19 \div 7 = 2 \text{ R } 5$$

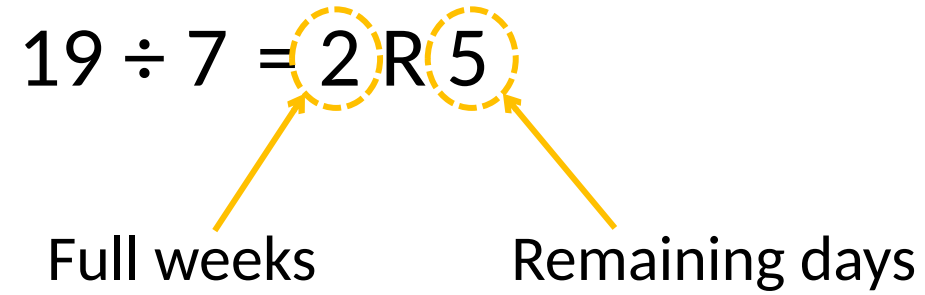
Full weeks Remaining days

The diagram illustrates the division of 19 by 7. The equation $19 \div 7 = 2 \text{ R } 5$ is shown. The quotient '2' is circled with a dashed orange line, and an orange arrow points from the text 'Full weeks' below to this circle. The remainder '5' is also circled with a dashed orange line, and an orange arrow points from the text 'Remaining days' below to this circle.

Weeks and Days

$$19 \div 7 = 2 \text{ R } 5$$

Full weeks Remaining days

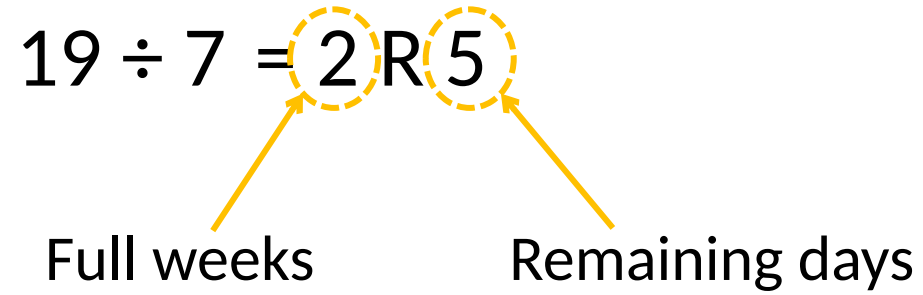


$$19 / 7 = 2$$

Weeks and Days

$$19 \div 7 = 2 \text{ R } 5$$

Full weeks Remaining days



$$19 / 7 = 2$$

$$19 \% 7 = 5$$