Cavalier Institute - https://cavalierinstitutions.com

Previous Assignment solutions

Assignment - 2 Solutions

Arrays

1. Input and print 5 numbers using an array

```c
#include <stdio.h>

int main() {
    int arr[5];
    printf("Enter 5 numbers:\n");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    printf("The entered numbers are:\n");
    for (int i = 0; i < 5; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

2. Find the largest number in an array of 5 integers

```c
#include <stdio.h>
```

```c
int main() {
    int arr[5], largest;
    printf("Enter 5 numbers:\n");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
    }
    largest = arr[0];
    for (int i = 1; i < 5; i++) {
        if (arr[i] > largest) {
            largest = arr[i];
        }
    }
    printf("The largest number is: %d\n", largest);
    return 0;
}
```

3. Calculate the sum of all elements in an array

```c
#include <stdio.h>

int main() {
    int arr[5], sum = 0;
    printf("Enter 5 numbers:\n");
    for (int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }
    printf("The sum of the elements is: %d\n", sum);
    return 0;
}
```

4. Store 10 integers and print only the even numbers

```c
#include <stdio.h>

int main() {
    int arr[10];
    printf("Enter 10 integers:\n");
    for (int i = 0; i < 10; i++) {
        scanf("%d", &arr[i]);
    }
    printf("Even numbers are:\n");
    for (int i = 0; i < 10; i++) {
        if (arr[i] % 2 == 0) {
            printf("%d ", arr[i]);
        }
    }
    return 0;
}
```

5. Count how many times the number 5 appears in an array of 10 elements

```c
#include <stdio.h>

int main() {
    int arr[10], count = 0;
    printf("Enter 10 numbers:\n");
    for (int i = 0; i < 10; i++) {
        scanf("%d", &arr[i]);
        if (arr[i] == 5) {
            count++;
        }
    }
    printf("Number 5 appears %d times.\n", count);
    return 0;
}
```

2D Arrays

1. Input and print a 2x2 matrix

```c
#include <stdio.h>

int main() {
    int matrix[2][2];
    printf("Enter 4 elements for a 2x2 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    printf("The 2x2 matrix is:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

2. Find the sum of all elements in a 2x2 matrix

```
#include <stdio.h>

int main() {
    int matrix[2][2], sum = 0;
    printf("Enter 4 elements for a 2x2 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix[i][j]);
            sum += matrix[i][j];
        }
    }
    printf("Sum of all elements: %d\n", sum);
    return 0;
}
```

3. Print the elements in row-major order

```
#include <stdio.h>

int main() {
    int matrix[2][2];
    printf("Enter 4 elements for a 2x2 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    printf("Row-major order:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            printf("%d ", matrix[i][j]);
        }
    }
    return 0;
}
```

4. Find the largest number in a 2x2 matrix

```
#include <stdio.h>

int main() {
    int matrix[2][2], largest;
    printf("Enter 4 elements for a 2x2 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    largest = matrix[0][0];
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            if (matrix[i][j] > largest) {
                largest = matrix[i][j];
            }
        }
    }
    printf("Largest number: %d\n", largest);
    return 0;
}
```

5. Addition of two 2x2 matrices

```
#include <stdio.h>

int main() {
    int A[2][2], B[2][2], result[2][2];
    printf("Enter elements for Matrix A:\n");
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            scanf("%d", &A[i][j]);

    printf("Enter elements for Matrix B:\n");
    for (int i = 0; i < 2; i++)
        for (int j = 0; j < 2; j++)
            scanf("%d", &B[i][j]);

    printf("Sum of Matrix A and B:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            result[i][j] = A[i][j] + B[i][j];
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Pointers

1. Print the value and address of an integer variable using a pointer

```
#include <stdio.h>

int main() {
    int a = 10;
    int *p = &a;

    printf("Value of a: %d\n", *p);
    printf("Address of a: %p\n", (void*)p);
    return 0;
}
```

2. Swap two numbers using pointers

```
#include <stdio.h>

void swap(int *x, int *y) {
    int temp = *x;
    *x = *y;
    *y = temp;
}

int main() {
    int a = 5, b = 10;
    printf("Before swapping: a = %d, b = %d\n", a, b);
    swap(&a, &b);
    printf("After swapping: a = %d, b = %d\n", a, b);
    return 0;
}
```

3. Increment a variable by 10 using a pointer

```
#include <stdio.h>

int main() {
    int a = 10;
    int *p = &a;

    *p += 10;
    printf("Value after incrementing by 10: %d\n", *p);

    return 0;
}
```

4. Display the elements of an array using a pointer

```
#include <stdio.h>

int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    int *p = arr;

    printf("Array elements are:\n");
    for (int i = 0; i < 5; i++) {
        printf("%d ", *(p + i));
    }

    return 0;
}
```

5. Calculate the sum of two numbers using pointers

```
#include <stdio.h>

int main() {
    int a, b, sum;
    int *p1 = &a, *p2 = &b;

    printf("Enter two numbers:\n");
    scanf("%d %d", p1, p2);

    sum = *p1 + *p2;
    printf("Sum: %d\n", sum);

    return 0;
}
```

Strings

1. Input and print a string

```
#include <stdio.h>

int main() {
    char str[100];
    printf("Enter a string:\n");
    gets(str);  // Input a string
    printf("You entered: %s\n", str);
    return 0;
}
```

2. Find the length of a string without `strlen()`

```
#include <stdio.h>

int main() {
    char str[100];
    int i, length = 0;

    printf("Enter a string:\n");
    gets(str);

    for (i = 0; str[i] != '\0'; i++) {
        length++;
    }

    printf("String length: %d\n", length);
    return 0;
}
```

3. Convert a string to uppercase

```
#include <stdio.h>

int main() {
    char str[100]; // vijeta\0
    printf("Enter a string:\n");
    gets(str);

    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] >= 'a' && str[i] <= 'z') {
            str[i] = str[i] - 32;
        }
    }

    printf("Uppercase string: %s\n", str);
    return 0;
}
```

`str[i] >= 'a'`: Checks if the character is greater than or equal to `'a'` (ASCII value 97).

`str[i] <= 'z'`: Checks if the character is less than or equal to `'z'` (ASCII value 122).

In the ASCII table, the difference between the lowercase letter and its corresponding uppercase letter is `32`. For example:

- `'a'` (ASCII 97) - `32` = `'A'` (ASCII 65)
- `'b'` (ASCII 98) - `32` = `'B'` (ASCII 66)

4. Compare two strings

```
#include <stdio.h>

int main() {
    char str1[100], str2[100];
    int i, vij = 0;

    printf("Enter first string:\n");
    gets(str1);
    printf("Enter second string:\n");
    gets(str2);

    for (i = 0; str1[i] != '\0' || str2[i] != '\0'; i++) {
        if (str1[i] != str2[i]) {
            vij = 1;
            break;
        }
    }

    if (flag == 0)
        printf("Strings are the same.\n");
    else
        printf("Strings are different.\n");

    return 0;
}
```

5. Count the number of vowels in a string

```
#include <stdio.h>
```

```
int main() {
    char str[100]; // vijetA rAj - ieaa 1234
    int count = 0;

    printf("Enter a string:\n");
    gets(str);

    for (int i = 0; str[i] != '\0'; i++) {
        if (str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i]
== 'o' || str[i] == 'u' ||
            str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i]
== 'O' || str[i] == 'U') {
            count++;
        }
    }

    printf("Number of vowels: %d\n", count);
    return 0;
}
```

Switch Statements

1. Display the day of the week

```
#include <stdio.h>

int main() {
    int day;
    printf("Enter a number (1-7): ");
    scanf("%d", &day);

    switch (day) {
        case 1: printf("Monday\n"); break;
        case 2: printf("Tuesday\n"); break;
        case 3: printf("Wednesday\n"); break;
        case 4: printf("Thursday\n"); break;
        case 5: printf("Friday\n"); break;
```

```c
        case 6: printf("Saturday\n"); break;
        case 7: printf("Sunday\n"); break;
        default: printf("Invalid input\n");
    }
    return 0;
}
```

2. Check if a character is a vowel or consonant

```c
#include <stdio.h>

int main() {
    char ch;
    printf("Enter a character: ");
    scanf(" %c", &ch);

    switch (ch) {
        case 'a': case 'e': case 'i': case 'o': case 'u':
        case 'A': case 'E': case 'I': case 'O': case 'U':
            printf("Vowel\n");
            break;
        default:
            printf("Consonant\n");
    }
    return 0;
}
```

3. Simple calculator

```c
#include <stdio.h>

int main() {
    char op;
    int a, b;

    printf("Enter operator (+, -, *, /): ");
    scanf(" %c", &op);

    printf("Enter two numbers: ");
    scanf("%d %d", &a, &b);
    switch (op) {
        case '+': printf("Result: %d\n", a + b); break;
        case '-': printf("Result: %d\n", a - b); break;
        case '*': printf("Result: %d\n", a * b); break;
        case '/':
            if (b != 0) printf("Result: %d\n", a / b);
            else printf("Division by zero error.\n");
            break;
        default: printf("Invalid operator.\n");
    }
    return 0;
}
```

4. Find if a number is positive, negative, or zero

```
#include <stdio.h>

int main() {
    int num;
    printf("Enter a number: ");
    scanf("%d", &num);

    switch ((num > 0) - (num < 0)) {
        case 1: printf("Positive\n"); break;
        case -1: printf("Negative\n"); break;
        case 0: printf("Zero\n"); break;
    }

    return 0;
}
Explanation :
```

`(num > 0)`:

- If `num` is greater than 0, this expression returns `1` (true).
- Otherwise, it returns `0` (false).

`(num < 0)`:

- If `num` is less than 0, this expression returns `1` (true).

- Otherwise, it returns `0` (false).

`(num > 0) - (num < 0)`:

- If `num > 0`, then `(1 - 0)` → **`1`** (positive).
- If `num < 0`, then `(0 - 1)` → **`-1`** (negative).
- If `num == 0`, then `(0 - 0)` → **`0`** (zero).

Assignment - 1 Solutions

Arrays

1. What is an array, and how is it different from a regular variable?

An **array** is a collection of elements of the same data type stored in contiguous memory locations.

- **Regular Variable**: Holds a single value.
- **Array**: Can hold multiple values, accessed using indices.

Example:

```
int x = 10;             // Regular variable

int arr[5] = {1, 2, 3, 4, 5};  // Array declaration
```

2. How do you declare and initialize an array in C? Provide an example.

```
#include <stdio.h>

int main() {

    int arr[5] = {1, 2, 3, 4, 5}; // Declaration and Initialization

    for (int i = 0; i < 5; i++) {

        printf("%d ", arr[i]);

    }

    return 0;
}
```

3. Write a program to find the largest number in a given array.

```
#include <stdio.h>

int main() {

    int arr[] = {10, 25, 5, 40, 15};

    int n = sizeof(arr) / sizeof(arr[0]);

    int largest = arr[0];

    for (int i = 1; i < n; i++) {

        if (arr[i] > largest) {

            largest = arr[i];

        }

    }

    printf("Largest number: %d\n", largest);

    return 0;

}
```

4. Write a program to calculate the sum of all elements in an array.

```
#include <stdio.h>

int main() {

    int arr[] = {1, 2, 3, 4, 5};

    int n = sizeof(arr) / sizeof(arr[0]);

    int sum = 0;

    for (int i = 0; i < n; i++) {
        sum += arr[i];

    }

    printf("Sum of elements: %d\n", sum);

    return 0;

}
```

5. How do you access the first and last elements of an array?

```
#include <stdio.h>

int main() {

    int arr[] = {10, 20, 30, 40, 50};

    int n = sizeof(arr) / sizeof(arr[0]);

    printf("First element: %d\n", arr[0]);

    printf("Last element: %d\n", arr[n-1]);

    return 0;

}
```

6. What happens if you try to access an element outside the bounds of an array?

Accessing an out-of-bounds element leads to **undefined behavior** in C, which may crash the program or return garbage values.

7. Write a program to reverse the elements of an array.

```
#include <stdio.h>

int main() {

    int arr[] = {1, 2, 3, 4, 5};

    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Reversed array: ");

    for (int i = n - 1; i >= 0; i--) {

        printf("%d ", arr[i]);

    }

    return 0;

}
```

8. How can you check if an array is empty?

In C, you need to keep track of the size explicitly. If `size = 0`, the array is empty.

9. Write a program to count the number of even and odd numbers in an array.

```
#include <stdio.h>

int main() {

    int arr[] = {1, 2, 3, 4, 5, 6};

    int n = sizeof(arr) / sizeof(arr[0]);

    int even = 0, odd = 0;

    for (int i = 0; i < n; i++) {

        if (arr[i] % 2 == 0)

            even++;

        else

            odd++;

    }

    printf("Even numbers: %d\n", even);

    printf("Odd numbers: %d\n", odd);
    return 0;

}
```

10. Explain the difference between a one-dimensional and two-dimensional array with examples.

- **One-Dimensional Array**:

```
int arr[5] = {1, 2, 3, 4, 5};
```

- **Two-Dimensional Array**:

```
int matrix[2][3] = {{1, 2, 3}, {4, 5, 6}};
```

Functions

1. What is a function, and why is it used?

A function is a block of code that performs a specific task. It improves modularity, reusability, and readability.

2. Write a function to calculate the square of a number.

```
#include <stdio.h>

int square(int num) {

    return num * num;

}

int main() {

    printf("Square of 5: %d\n", square(5));

    return 0;

}
```

3. How do you pass arguments to a function? Explain with an example.

```
#include <stdio.h>
```

```
void greet(char name[]) {

    printf("Hello, %s\n", name);

}

int main() {

    greet("Alice");

    return 0;

}
```

4. What is the difference between passing arguments by value and by reference?

- **By Value**: A copy of the variable is passed.
- **By Reference**: The address of the variable is passed.

Example (By Reference):

```
#include <stdio.h>

void modify(int *num) {

    *num = 10;

}

int main() {

    int x = 5;

    modify(&x);

    printf("Modified value: %d\n", x);

    return 0;

}
```

5. Write a function to check if a number is prime.

```c
#include <stdio.h>

#include <stdbool.h>

bool isPrime(int n) {

    if (n <= 1) return false;

    for (int i = 2; i <= n / 2; i++) {

        if (n % i == 0) return false;

    }

    return true;

}

int main() {

    int num = 7;

    if (isPrime(num))

        printf("%d is Prime\n", num);

    else

        printf("%d is not Prime\n", num);

    return 0;

}
```

6. What is a return statement, and how is it used?

The `return` statement sends a value back to the caller function.

7. Can a function return multiple values?

In C, you can use pointers or structures to return multiple values.

8. Write a function to find the factorial of a number.

```
#include <stdio.h>

int factorial(int n) {

    if (n == 0 || n == 1) return 1;

    return n * factorial(n - 1);

}

int main() {

    printf("Factorial of 5: %d\n", factorial(5));

    return 0;

}
```

9. How can you call a function inside another function? Provide an example.

```
#include <stdio.h>

int square(int n) {

    return n * n;

}

int cube(int n) {

    return n * square(n);

}

int main() {

    printf("Cube of 3: %d\n", cube(3));

    return 0;

}
```

END