**Experiment- 11**

|  |
| --- |
| **Student Name and Roll Number:** |
| **Semester /Section: First Sem.** |
| **Link to Code:** |
| **Date: 13 Nov. 2024** |
| **Faculty Signature:** |

|  |
| --- |
| **Objective**  To familiarize the students with switch-case statements. |
| **Program Outcome**  The students will learn the concept of looping in C. They will be able to understand where and how switch case statements are implemented. |
| **Problem Statement**  1. Write a C Program with an algorithm and flowchart to make a simple calculator performing +,-,/,\* using switch-case. The program takes two integer operands and one operator from the user, performs the operation and then prints the result.  **Code:**  **Output and screenshot**  2. A character is entered through keyboard. Write a C program and its algorithm to determine whether the character entered is a capital letter, a small case letter, a digit or a special symbol using switch case. The following table shows the range of ASCII values for various characters.  **Characters ASCII values**  A – Z 65 – 90  a – z 97 – 122  0 – 9 48 – 57  Special symbols 0 – 47, 58 – 64, 91 – 96, 123 – 127  **Code:**  **Output and screenshot**      3. Write a C program using switch case to display grade of students based on the total marks obtained by the student in five subjects  Here is the range of Grades:  Marks >= 90 : Grade A  Marks >= 70 && < 90 : Grade B  Marks >= 50 && < 70 : Grade C  Marks < 50 : Grade D  Marks<50 : Fail  *e.g. Program Outcome*  Enter marks of 5 subjects  97 89 78 87 68  Grade : B |
| **Background**  The switch statement allows us to execute one code block among many alternatives.  You can do the same thing with the if...else..if ladder. However, the syntax of the switch statement is much easier to read and write.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Syntax of switch...case  switch (expression)  {  case constant1:  // statements  break;  case constant2:  // statements  break;  .  .  .  default:  // default statements  }  *How does the switch statement work?*  The expression is evaluated once and compared with the values of each case label.   * If there is a match, the corresponding statements after the matching label are executed. For example, if the value of the expression is equal to constant2, statements after case constant2: are executed until break is encountered. * If there is no match, the default statements are executed. * If we do not use break, all statements after the matching label are executed. * The default clause inside the switch statement is optional.   *Rules for switch-case*   1. The expression provided in the switch should result in a constant value otherwise it would not be valid.   Valid expressions for switch:  // Constant expressions allowed  switch(1+2+23)  switch(1\*2+3%4)  // Variable expression are allowed provided  // they are assigned with fixed values  switch(a\*b+c\*d)  switch(a+b+c)   1. Duplicate case values are not allowed. 2. The default statement is optional. Even if the switch case statement do not have a default statement, it would run without any problem. 3. The break statement is used inside the switch to terminate a statement sequence. When a break statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement. 4. The break statement is optional. If omitted, execution will continue on into the next case. The flow of control will fall through to subsequent cases until a break is reached. 5. Nesting of switch statements are allowed, which means you can have switch statements inside another switch. However nested switch statements should be avoided as it makes program more complex and less readable. |
| **Question Bank**  1. Why do we use break keyword ?  2. Can we assign float value to a selection variable in switch?  3. In what ways does a switch statement differ from if statement?  4. Which of the following statement is correct for switch controlling expression?   1. Only int can be used in “switch” control expression 2. Both int and char can be used in “switch” control expression. 3. All types i.e. int, char and float can be used in “switch” control expression 4. “switch” control expression can be empty as well |
| **Flipped Questions**  1. What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)  #include <stdio.h>  void main()  {  double ch;  printf("enter a value between 1 to 2:");  scanf("%lf", &ch);  switch (ch)  {  case 1:  printf("1");  break;  case 2:  printf("2");  break;  }  }  2) What will be the output of the following C code? (Assuming that we have entered the value 1 in the standard input)  #include <stdio.h>  void main()  {  int ch;  printf("enter a value between 1 to 2:");  scanf("%d", &ch);  switch (ch)  {  case 1:  printf("1\n");  default:  printf("2\n");  }  }  3) What will be the output of the following C code?  #include <stdio.h>  int main()  {  int a = 1, b = 1;  switch (a)  {  case a\*b:  printf("yes ");  case a-b:  printf("no\n");  break;  }  }  4) What is the output of the below program?   |  | | --- | | #include <stdio.h>  int main()  {  int i = 0;  switch (i)  {  case '0': printf("Geeks");  break;  case '1': printf("Quiz");  break;  default: printf("GeeksQuiz");  }  return 0;  } |   5) What will be the output of the following C program segment? (GATE CS 2012)   |  | | --- | | char inchar = 'A';  switch (inchar)  {  case 'A' :  printf ("choice A n") ;  case 'B' :  printf ("choice B ") ;  case 'C' :  case 'D' :  case 'E' :  default:  printf ("No Choice") ;  } | |

**Solutions to Problem Statements:**

**1. Code:**

#include <stdio.h>

int main() {

int num1, num2;

char operator;

float result;

printf("Enter first number: ");

scanf("%d", &num1);

printf("Enter an operator (+, -, \*, /): ");

scanf(" %c", &operator);

printf("Enter second number: ");

scanf("%d", &num2);

switch (operator) {

case '+':

result = num1 + num2;

break;

case '-':

result = num1 - num2;

break;

case '\*':

result = num1 \* num2;

break;

case '/':

if (num2 != 0) {

result = (float)num1 / num2;

} else {

printf("Error: Division by zero is not allowed.\n");

return 1;

}

break;

default:

printf("Invalid operator.\n");

return 1;

}

printf("Result: %.2f\n", result);

return 0;

}

**Screenshot:**

A screenshot of a computer

Description automatically generated

**2. Code:**

#include <stdio.h>

int main() {

char ch;

int asciiValue;

printf("Enter a character: ");

scanf("%c", &ch);

asciiValue = (int)ch;

switch (1) {

case 1:

if (asciiValue >= 65 && asciiValue <= 90) {

printf("The character is a capital letter.\n");

break;

}

case 2:

if (asciiValue >= 97 && asciiValue <= 122) {

printf("The character is a lowercase letter.\n");

break;

}

case 3:

if (asciiValue >= 48 && asciiValue <= 57) {

printf("The character is a digit.\n");

break;

}

case 4:

if ((asciiValue >= 0 && asciiValue <= 47) ||

(asciiValue >= 58 && asciiValue <= 64) ||

(asciiValue >= 91 && asciiValue <= 96) ||

(asciiValue >= 123 && asciiValue <= 127)) {

printf("The character is a special symbol.\n");

break;

}

default:

printf("Invalid input.\n");

break;

}

return 0;

}

**Screenshot:**

**A screenshot of a computer

Description automatically generated**

**3. Code:**

#include <stdio.h>

int main() {

int marks[5], total = 0;

float average;

char grade;

printf("Enter marks for 5 subjects: ");

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

scanf("%d", &marks[i]);

total += marks[i];

}

average = total / 5.0;

switch (1) {

case 1:

if (average >= 90) {

grade = 'A';

break;

}

case 2:

if (average >= 70 && average < 90) {

grade = 'B';

break;

}

case 3:

if (average >= 50 && average < 70) {

grade = 'C';

break;

}

case 4:

if (average < 50) {

grade = 'D';

break;

}

default:

grade = 'F';

break;

}

if (grade == 'D') {

printf("Grade: %c (Fail)\n", grade);

} else {

printf("Grade: %c\n", grade);

}

return 0;

}

**Screenshot:**

**A screenshot of a computer

Description automatically generated**

**Solution to Question Bank:**

**1.** We use the `break` keyword to exit from loops or `switch` statements immediately. This prevents the program from executing further code within the loop or `switch`, allowing us to control when to stop or move to the next part of the program.

**2.** No, we cannot use a `float` value as the selection variable in a `switch` statement. In C, the `switch` statement only accepts integer or character types (such as `int`, `char`, `short`, or `long`) for the selection variable. This is because `switch` relies on exact matches, and `float` values can have precision issues, making them unreliable for this purpose.

**3.** Differences between switch and if statements:

- Type of Conditions: if can handle complex conditions, including ranges and logical operators, while switch is limited to discrete, single-value cases.

- Data Types: switch only accepts integer or character data types for its control expression, while if can evaluate any data type or expression.

- Execution Flow: switch evaluates only one expression and matches against case labels, while if evaluates multiple conditions in sequence.

- Readability: switch is often more readable for handling multiple fixed values, whereas if is more flexible for complex logic.

**3.** b) Both int and char can be used in “switch” control expression.

**Solution to flipped questions:**

**1. Output:** There will be a compilation error.

**Screenshot:**

**A screenshot of a computer program

Description automatically generated**

**2. Output:**

enter a value between 1 to 2:1

1

2

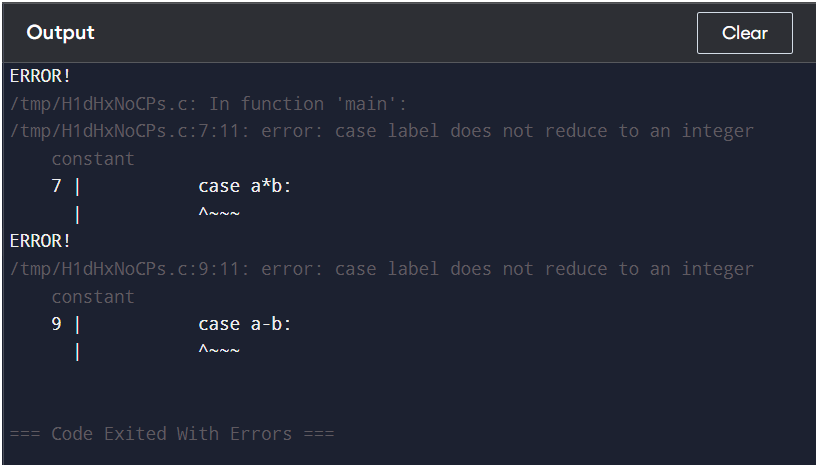
**Screenshot:**

**A screenshot of a computer

Description automatically generated**

**3. Output:** Compilation Error: The error is caused by the case labels in the switch statement**.**

**Screenshot:**

****

**4. Output:** GeeksQuiz

**Screenshot:**

**A screen shot of a computer

Description automatically generated**

**5. Output:** choice A nchoice B No Choice

**Screenshot:**

**A black rectangular object with a black border

Description automatically generated with medium confidence**

**Experiment- 12**

|  |
| --- |
| **Student Name and Roll Number:** |
| **Semester /Section: First Sem.** |
| **Link to Code:** |
| **Date: 16 Nov. 2024** |
| **Faculty Signature:** |

|  |
| --- |
| **Objective**  To familiarize the students with functions;call by value and call by reference. |
| **Program Outcome**  The students will learn the concept of functions in C. They will be able to understand where and how different functions are implemented. |
| **Problem Statement**  1. Write a C Program to print “Welcome to C programming“ using function.  **Code:**  #include<stdio.h>  int printMessage();  int main() {  printMessage();  return 0;  }  void printMessage() {  printf("Welcome to C programming\n");  }  **Output and screenshot :**     1. Write a C function to explain the concept of non-parameterized and parameterized functions.   **Code:**  #include <stdio.h>  // Non-parameterized function  void greet() {  printf("Hello, welcome to C programming!\n");  }  // Parameterized function  void displayNumber(int num) {  printf("The number is: %d\n", num);  }  int main() {  // Calling non-parameterized function  greet();  // Calling parameterized function  displayNumber(5);  return 0;  }  **Output and screenshot:**    3. Write a C program to calculate the sum of 3 numbers. The user enters the numbers, and the function should return the sum calculated to the main function.  **Code:**  #include <stdio.h>  // Function to calculate the sum of three numbers  int calculateSum(int num1, int num2, int num3) {  return num1 + num2 + num3;  }  int main() {  int num1, num2, num3, sum;  // Taking user input  printf("Enter three numbers: ");  scanf("%d %d %d", &num1, &num2, &num3);  // Calling the function to calculate the sum  sum = calculateSum(num1, num2, num3);  // Displaying the result  printf("The sum of %d, %d, and %d is: %d\n", num1, num2, num3, sum);  return 0;  }  **Output and screenshot:**     1. Write a C program to swap two numbers. Implement using call by value and call by reference and explain any difference observed in the output.   **Code:**  #include <stdio.h>  // Function to swap using Call by Value  void swapByValue(int a, int b) {  int temp = a;  a = b;  b = temp;  printf("In swapByValue: a = %d, b = %d\n", a, b);  }  // Function to swap using Call by Reference  void swapByReference(int \*a, int \*b) {  int temp = \*a;  \*a = \*b;  \*b = temp;  printf("In swapByReference: a = %d, b = %d\n", \*a, \*b);  }  int main() {  int num1, num2;  // Taking user input  printf("Enter two numbers: ");  scanf("%d %d", &num1, &num2);  // Swap using Call by Value  printf("Before swapByValue: num1 = %d, num2 = %d\n", num1, num2);  swapByValue(num1, num2);  printf("After swapByValue: num1 = %d, num2 = %d\n", num1, num2);  // Swap using Call by Reference  printf("\nBefore swapByReference: num1 = %d, num2 = %d\n", num1, num2);  swapByReference(&num1, &num2);  printf("After swapByReference: num1 = %d, num2 = %d\n", num1, num2);  return 0;  }  **Output and screenshot:**     1. Write a C program to calculate the square of a number using functions.   **Code:**  #include <stdio.h>  int calculateSquare(int num) {  return num \* num;  }  int main() {  int number, result;  printf("Enter a number: ");  scanf("%d", &number);  result = calculateSquare(number);  printf("The square of %d is %d\n", number, result);  return 0;  }  **Output and screenshot:** |
|  |

**Experiment- 13**

|  |
| --- |
| **Student Name and Roll Number:** |
| **Semester /Section: First** |
| **Link to Code:** |
| **Date: 17/11/2024** |
| **Faculty Signature:** |

|  |
| --- |
| **Objective**  To familiarize the students with arrays. |
| **Program Outcome**  The students will learn the concept of arrays in C. They will be able to understand where and how arrays are implemented. |
| **Problem Statement**  1. Write a program that takes 10 numbers as input, stores them in an array, and prints all the elements.  **Code:**  #include <stdio.h>  int main() {  int numbers[10];  printf("Enter 10 numbers:\n");  for (int i = 0; i < 10; i++) {  printf("Enter number %d: ", i + 1);  scanf("%d", &numbers[i]);  }    printf("\nThe entered numbers are:\n");  for (int i = 0; i < 10; i++) {  printf("%d ", numbers[i]);  }  printf("\n");  return 0;  }  **Output and screenshot**    2. Write a C program to calculate students' percentage based on their marks in five subjects. The user will input the marks for each of the five subjects, which will be stored in an array. The program will then compute and display the student's percentage.  **Code:**  #include <stdio.h>  int main() {  float marks[5], total = 0, percentage;  printf("Enter the marks for 5 subjects:\n");  for (int i = 0; i < 5; i++) {  printf("Subject %d: ", i + 1);  scanf("%f", &marks[i]);    total += marks[i];  }    percentage = (total / 500) \* 100;  printf("\nTotal Marks: %.2f\n", total);  printf("Percentage: %.2f%%\n", percentage);  return 0;  }  **Output and screenshot**       1. Write a program that takes 10 numbers as input, stores them in an array, and prints all the elements using a function.   **Code:**  #include <stdio.h>  // Function to print all elements of the array  void printArray(int arr[], int size) {  for (int i = 0; i < size; i++) {  printf("%d ", arr[i]);  }  printf("\n");  }  int main() {  int numbers[10];  // Input 10 numbers  printf("Enter 10 numbers:\n");  for (int i = 0; i < 10; i++) {  scanf("%d", &numbers[i]);  }  // Print the array elements  printf("The entered numbers are:\n");  printArray(numbers, 10);  return 0;  }  **Output and screenshot**     1. Write a C program to copy elements from one array into another array and display the copied array.   **Code:**  #include <stdio.h>  int main() {  int original[5], copy[5];  // Input elements for the original array  printf("Enter 5 elements for the original array:\n");  for (int i = 0; i < 5; i++) {  scanf("%d", &original[i]);  }  // Copy elements from original array to copy array  for (int i = 0; i < 5; i++) {  copy[i] = original[i];  }  // Display the copied array  printf("The copied array is:\n");  for (int i = 0; i < 5; i++) {  printf("%d ", copy[i]);  }  printf("\n");  return 0;  }  **Output and screenshot**     1. Write a C program to find the difference between the largest and smallest elements in an array of integers.   **Code:**  #include <stdio.h>  int main() {  int arr[5], largest, smallest;  // Input elements for the array  printf("Enter 5 integers:\n");  for (int i = 0; i < 5; i++) {  scanf("%d", &arr[i]);  }  // Initialize largest and smallest with the first element  largest = smallest = arr[0];  // Find the largest and smallest elements  for (int i = 1; i < 5; i++) {  if (arr[i] > largest) largest = arr[i];  if (arr[i] < smallest) smallest = arr[i];  }  // Display the result  printf("Difference: %d\n", largest - smallest);  return 0;  }  **Output and screenshot** |

**Experiment- 14**

|  |
| --- |
| **Student Name and Roll Number:** |
| **Semester /Section: First** |
| **Link to Code:** |
| **Date: 17/11/2024** |
| **Faculty Signature:** |

|  |
| --- |
| **Objective**  To familiarize the students with srings. |
| **Program Outcome**  The students will learn the concept of strings in C. They will be able to understand where and how strings are implemented. |
| **Problem Statement**  1. Write a C program to display your name as a string  **Code:**  **Output and screenshot**  2.Write a C program that checks if a given string is a palindrome (reads the same forward and backward).  **Code:**  #include <stdio.h>  int main() {  char name[50];  // Correct usage without the '&'  printf("Enter your name: ");  scanf("%s", name); // 'name' is already a pointer to the array  printf("Your name is: %s\n", name);  return 0;  }  **Output and screenshot:**     1. Write a C program to reverse the string.   **Code:**  **Output and screenshot**       1. Write a C program to count the number of words in a string.   **Code:**  #include <stdio.h>  int main() {  char str[100];  int count = 0;  // Input the string  printf("Enter a string: ");  gets(str); // gets() allows input with spaces  // Count words  for (int i = 0; str[i] != '\0'; i++) {  if (str[i] != ' ' && (i == 0 || str[i-1] == ' ')) {  count++;  }  }  // Display the word count  printf("Number of words: %d\n", count);  return 0;  }  **Output and screenshot**     1. Write a C program to concatenate (join) two strings .   **Code:**  #include <stdio.h>  int main() {  char str1[100], str2[100];  // Input two strings  printf("Enter first string: ");  fgets(str1, sizeof(str1), stdin);  printf("Enter second string: ");  fgets(str2, sizeof(str2), stdin);  // Remove the newline character from both strings  if (str1[strlen(str1) - 1] == '\n') {  str1[strlen(str1) - 1] = '\0';  }  if (str2[strlen(str2) - 1] == '\n') {  str2[strlen(str2) - 1] = '\0';  }  // Concatenate str2 to str1  int i = 0;  while (str1[i] != '\0') {  i++;  }  int j = 0;  while (str2[j] != '\0') {  str1[i] = str2[j];  i++;  j++;  }  str1[i] = '\0'; // Null-terminate the concatenated string  // Display the concatenated string  printf("Concatenated string: %s\n", str1);  return 0;  }  **Output and screenshot**     1. Write a C program to copy one string to other string.   **Code:**  #include <stdio.h>  int main() {  char str1[100], str2[100];  int i = 0;  // Input the first string  printf("Enter the string to copy: ");  scanf("%[^\n]s", str1); // Reads a line of input including spaces  // Copy str1 to str2  while (str1[i] != '\0') {  str2[i] = str1[i];  i++;  }  str2[i] = '\0'; // Null-terminate the second string  // Display the copied string  printf("Copied string: %s\n", str2);  return 0;  }  **Output and screenshot:** |