Assignment 2

Question 1:

```
#include <stdio.h> //including the general input and the output header file
int main(){   //Declaring the main function which is single in a given file
   int a,b;     //Declaring a as the integer variable
   printf("Enter the number: ");
   scanf("%d",&a);      //Taking the input from the user
   b = a%10;
   printf("The unit digit of the number %d is %d",a,b);
   return 0;       //Telling the os that the program has successfully executed

}
```

Question 2:

```
#include <stdio.h>  //Including the standard input and the output header file
int main(){   //Declaring the main function which is single in a file
   int a,b;
   printf("Enter the number: ");
   scanf("%d",&a);
   b = a/10;
   printf("The number %d without the last digit is %d",a,b);
   return 0;   //Telling the os that the program has successfully executed

}
```

Question 3:

```
#include <stdio.h>  //Including the standard input and the output header file
int main(){  //Creating the main function which is single in a file
   int a,b,t;
   printf("Enter the first number: ");
   scanf("%d",&a);
```

```
   printf("Enter the second number: ");
   scanf("%d",&b);
   printf("Initially the number a=%d and b=%d\n",a,b);
   t=a;
   a=b;
   b=t;
   printf("Finally numbers are a=%d and b=%d",a,b);
   return 0;    //Telling the os that the program has successfully executed
}
```

Question 4: Swapping the numbers using the third variable

```
#include <stdio.h>  //Including the standard input and the output header file
int main(){  //Creating the main function which is single in a file
   int a,b,t;
   printf("Enter the first number: ");
   scanf("%d",&a);
   printf("Enter the second number: ");
   scanf("%d",&b);
   printf("Initially the number a=%d and b=%d\n",a,b);
   t=a;
   a=b;
   b=t;
   printf("Finally numbers are a=%d and b=%d",a,b);
   return 0;    //Telling the os that the program has successfully executed
}
```

Question 4: Swapping the numbers without using the third variable

```
#include <stdio.h> //including the standard input and the output header file
int main(){  //Declaring the main function which can be declared only once in a file
   int a,b;
   printf("Enter the first number: ");
```

```
   scanf("%d",&a);
   printf("Enter the second number: ");
   scanf("%d",&b);
   printf("Before swapping the numbers are a=%d and b=%d\n",a,b);
   a=a+b;
   b=a-b;
   a=a-b;
   printf("After swapping the numbers are a=%d and b=%d",a,b);
   return 0;    //Telling the os that the program has successfully executed
}
```

Question 5:

```
#include <stdio.h>  //including the standard input and the output header file
int main(){
   int number,digit1,digit2,digit3,sum;
   printf("Enter a three digit number: ");
   scanf("%d",&number);
   digit1 = number/100;  //Extracts the hundreds digit of a number
   digit2 = (number/10)%10; //Extracts the tens digit of a number
   digit3 =  number%10;     //Extracts the ones digit of a number
   sum = digit1+digit2+digit3;
   printf("The sum of the digits of a three digit number %d is %d",number,sum);
   return 0;
}
```

Question 6:

```
#include <stdio.h>  //Including the standard input and output header file
int main(){    //Declaring the main function which is only one in the entire main function
   char a;  //Declaring a as the character input
   printf("Enter a character: ");
   scanf("%c",&a);   //Taking the character input
```

```
   printf("The ASCII code of '%c' is: %d",a,a);  //printing the ASCII value
   return 0;   //Telling the os that the program has successfully done
}
```

Question 7:

```
#include <stdio.h>  //Including the standard input and the output header file
int main(){        //Declaring the main function which is only one in the given file
   int a;         //Declaring a as the integer variable
   printf("Enter the number: ");
   scanf("%d",&a);
   if(a & 1){
      printf("Odd number");
   }
   else{
      printf("Even number");
   }
   return 0;   //Telling the os that the program has successfully executed
}
```

Question 8:

```
#include <stdio.h>
int main(){
   printf("Size of int: %zu bytes\n",sizeof(int));
   printf("Size of float: %zu bytes\n",sizeof(float));
   printf("Size of char: %zu bytes\n",sizeof(char));
   printf("Size of double: %zu bytes\n",sizeof(double));
   return 0;
}
```

Explanation:

The `%zu` format specifier in C is used for printing the result of the `sizeof()` operator and other size-related values. Let's break down its meaning:

- `%`: This is the format specifier character used in `printf()` and `scanf()` functions to indicate that a placeholder for a value is being used in the format string.

- `z`: This is a length modifier that is specific to `printf()` and `scanf()`. In `%zu`, the 'z' modifier indicates that the corresponding argument should be treated as a `size_t` data type. `size_t` is an unsigned integer type used for representing sizes, such as the result of the `sizeof()` operator or array indices. The 'z' modifier ensures that the correct size is used, which can vary depending on the system's architecture.

- `u`: This stands for "unsigned." It specifies that the argument being printed or scanned is an unsigned integer.

So, when you use `%zu` in a `printf()` statement, you are telling the function to expect an argument of type `size_t` (an unsigned integer representing a size) and to print it as an unsigned integer. This is particularly useful when you want to print the size of data types or the results of size-related operations like `sizeof()`, ensuring that the output is correctly formatted and platform-independent.

The `%zu` format specifier is used in C (and C++) to correctly print the result of the `sizeof()` operator, which gives the size in bytes of a data type or an expression. It is important to use the correct format specifier for `sizeof()` because the size of data types can vary across different systems and compilers, and `sizeof()` returns an unsigned integer type (`size_t`), which may have a different size depending on the system architecture.

Here's why `%zu` is used:

1. **Portability**: Using `%zu` ensures that the code is portable and works correctly on different systems. Some systems might have `size_t` as a 32-bit unsigned integer, while others might have it as a 64-bit unsigned integer. Using the correct format specifier ensures that the size is printed accurately, regardless of the platform.

2. **Type Mismatch**: If you use the wrong format specifier (e.g., `%d` for `sizeof()`), it can lead to type mismatch issues and undefined behavior because `sizeof()` returns an unsigned integer type (`size_t`), and using a format specifier for a signed integer type can result in incorrect output or even crashes.

3. **Compiler Warnings**: Modern compilers often generate warnings if you use the incorrect format specifier for `sizeof()`, helping you catch potential issues in your code.

Here's an example of how `%zu` is used:

```c
#include <stdio.h>

int main() {
    printf("Size of int: %zu bytes\n", sizeof(int));
    printf("Size of double: %zu bytes\n", sizeof(double));

    return 0;
}
```

In this code, `%zu` is used to correctly print the size of the `int` and `double` data types in bytes. The output will vary depending on the system, but `%zu` ensures that it's displayed accurately.

Question 9:

```
#include <stdio.h>  //including the standard input and the output header file
int main(){     //Creating the main function which is created only once in a given file
    int a;
    printf("Enter the number: ");  //Using the double quotes because of the string constant
    scanf("%d",&a);
    a = (a/10)*10;
    printf("The required number is: %d",a);
    return 0;

}
```

Question 10: //using the string inputs

```
#include <stdio.h>  //including the standard input and the output header file
#include <string.h>
int main(){        //including the standard input and the output header file
   char a[20];     //creating the character array
   char b[20];
   char c[40];
   printf("Enter the number: ");
   scanf("%s",a);
   printf("Enter the digit to be appended: ");
   scanf("%s",b);
   strcpy(c,a);
   strcat(c,b);
   printf("The new number is: %s",c);
   return 0;   //Telling the os that the program has successfully executed
}
```

Question 10.1 //Using the integer input

```
#include <stdio.h>  //including the standard input and the output header file
int main(){    //Creating the main function which is created only once in a file
   int a,b,c;
   printf("Enter the number: ");
   scanf("%d",&a);
   printf("Enter the second number: ");
   scanf("%d",&b);
   c = (a*10)+b;
   printf("The new number is: %d",c);
   return 0;       //Telling the os that the program has successfully executed
}
```

Question 11

```
#include <stdio.h>  //including the standard input and the output header file
int main(){     //Creating the main function which is created only once in the file
   float a,b;
   printf("Enter the amount in rupees: ");
   scanf("%f",&a);
   b = a/76.23;
   printf("The price in USD is %.3f",b);
   return 0;  //Telling the os that the program has successfully executed
}
```

Question 12

```
#include <stdio.h>   //including the standard input and output header fie
int main(){         //creating the main function which is created only once in a file
   int x,a,b;       //Declaring x,a,b as the integer variables
   printf("Enter a three digit number: ");  //Using the double quotes because of the string
constant
   scanf("%d",&x);
   a=x%10;          //Extracting the ones digit
   x=x/10;          //removing the ones digit
   b=(a*100)+x;
   printf("The required number is: %d",b);
   return 0;      //Telling the os that the program has successfully executed
}
```

Question 12.1

```
#include <stdio.h>  //including the standard input and output header file
int main(){         //creating the main function which is created only once in a file
   int a,b,c,d,e;
   printf("Enter a three digit number: ");
   scanf("%d",&a);
   b=a%10;       //Extracting the ones digit
```

```
    c=(a/10)%10;   //extracting the tens digit
    d=(a/100)%10;  //extracting the hundreds digit
    e = ((b*100)+(d*10)+c);
    printf("The required number is: %d",e);
    return 0;      //Telling the os that the program has successfully executed
}
```