Date: 06 Aug  
Lab 1 DSA  
  
Question 1: Create an array of integers. Using pointer arithmetic, find the sum of all elements in the array without using array indexing  
  
code:  
#include <stdio.h>

int main() {

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

int length = sizeof(array) / sizeof(array[0]);

int sum = 0;

int \*ptr = array;

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

sum += \*(ptr + i);

}

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

return 0;

}  
  
output:  
Sum of all elements: 15  
  
  
question 2: Create a structure to represent a complex number. Write functions for adding, subtracting, and multiplying complex numbers

a.Adding by using call by value

Subtracting and Multiplying by using call by reference  
  
code:  
#include <stdio.h>

typedef struct {

float real;

float imag;

} Complex;

Complex addComplex(Complex c1, Complex c2) {

Complex result;

result.real = c1.real + c2.real;

result.imag = c1.imag + c2.imag;

return result;

}

void subtractComplex(Complex \*c1, Complex \*c2, Complex \*result) {

result->real = c1->real - c2->real;

result->imag = c1->imag - c2->imag;

}

void multiplyComplex(Complex \*c1, Complex \*c2, Complex \*result) {

result->real = c1->real \* c2->real - c1->imag \* c2->imag;

result->imag = c1->real \* c2->imag + c1->imag \* c2->real;

}

int main() {

Complex c1 = {3.0, 2.0};

Complex c2 = {1.0, 7.0};

Complex sum = addComplex(c1, c2);

printf("Sum: %.2f + %.2fi\n", sum.real, sum.imag);

Complex difference;

subtractComplex(&c1, &c2, &difference);

printf("Difference: %.2f + %.2fi\n", difference.real, difference.imag);

Complex product;

multiplyComplex(&c1, &c2, &product);

printf("Product: %.2f + %.2fi\n", product.real, product.imag);

return 0;

}  
  
output:  
Sum: 4.00 + 9.00i

Difference: 2.00 + -5.00i

Product: -11.00 + 23.00i  
  
  
Question 3:  
3.Create a structure to represent a book (title, author, price). Dynamically allocate memory for an array of books. Take the input from the user and store the details in the array. Write a function to print the book details.  
  
code:  
#include <stdio.h>

#include <stdlib.h>

typedef struct {

char title[100];

char author[100];

float price;

} Book;

void printBookDetails(Book \*books, int n) {

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

printf("Book %d:\n", i + 1);

printf("Title: %s\n", books[i].title);

printf("Author: %s\n", books[i].author);

printf("Price: %.2f\n", books[i].price);

printf("\n");

}

}

int main() {

int n;

printf("Enter the number of books: ");

scanf("%d", &n);

Book \*books = (Book \*)malloc(n \* sizeof(Book));

if (books == NULL) {

printf("Memory allocation failed\n");

return 1;

}

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

printf("Enter details for book %d:\n", i + 1);

printf("Title: ");

scanf(" %[^\n]%\*c", books[i].title);

printf("Author: ");

scanf(" %[^\n]%\*c", books[i].author);

printf("Price: ");

scanf("%f", &books[i].price);

printf("\n");

}

printBookDetails(books, n);

free(books);

return 0;

}  
  
output:  
Enter the number of books: 2

Enter details for book 1:

Title: DSA

Author: siddhi

Price: 200

Enter details for book 2:

Title: DAA

Author: siddhi Nyati

Price: 300

Book 1:

Title: DSA

Author: siddhi

Price: 200.00

Book 2:

Title: DAA

Author: siddhi Nyati

Price: 300.00  
  
  
Question 4:  
4.Write a program to perform the following:

a.Create an array of 5 student structures, each containing name, roll number, and marks.

b.Write functions to return the student with the highest marks by

i. call by value  
  
code:  
#include <stdio.h>

typedef struct {

char name[100];

int rollNumber;

float marks;

} Student;

Student getStudentWithHighestMarks(Student students[], int n) {

Student topStudent = students[0];

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

if (students[i].marks > topStudent.marks) {

topStudent = students[i];

}

}

return topStudent;

}

int main() {

Student students[2];

for (int i = 0; i < 2; i++) {

printf("Enter details for student %d:\n", i + 1);

printf("Name: ");

scanf(" %[^\n]%\*c", students[i].name);

printf("Roll Number: ");

scanf("%d", &students[i].rollNumber);

printf("Marks: ");

scanf("%f", &students[i].marks);

printf("\n");

}

Student topStudent = getStudentWithHighestMarks(students, 2);

printf("Student with the highest marks:\n");

printf("Name: %s\n", topStudent.name);

printf("Roll Number: %d\n", topStudent.rollNumber);

printf("Marks: %.2f\n", topStudent.marks);

return 0;

}  
  
output:  
Enter details for student 1:

Name: siddhi

Roll Number: 101

Marks: 90

Enter details for student 2:

Name: melisa

Roll Number: 102

Marks: 89

Student with the highest marks:

Name: siddhi

Roll Number: 101

Marks: 90.00

ii.Call by reference.  
  
code:

#include <stdio.h>

typedef struct {

char name[100];

int rollNumber;

float marks;

} Student;

void getStudentWithHighestMarks(Student students[], int n, Student \*topStudent) {

\*topStudent = students[0];

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

if (students[i].marks > topStudent->marks) {

\*topStudent = students[i];

}

}

}

int main() {

Student students[2];

for (int i = 0; i < 2; i++) {

printf("Enter details for student %d:\n", i + 1);

printf("Name: ");

scanf(" %[^\n]%\*c", students[i].name);

printf("Roll Number: ");

scanf("%d", &students[i].rollNumber);

printf("Marks: ");

scanf("%f", &students[i].marks);

printf("\n");

}

Student topStudent;

getStudentWithHighestMarks(students, 2, &topStudent);

printf("Student with the highest marks:\n");

printf("Name: %s\n", topStudent.name);

printf("Roll Number: %d\n", topStudent.rollNumber);

printf("Marks: %.2f\n", topStudent.marks);

return 0;

}  
  
  
Question 6.Dynamic allocate memory for a one-dimensional array that stores your input string. Write a program which implements all the given string functions

a.Finding length of a string:  
  
code:

#include <stdio.h>

#include <stdlib.h>

int stringLength(char \*str) {

int length = 0;

while (str[length] != '\0') {

length++;

}

return length;

}

int main() {

char \*str;

int size;

printf("Enter the maximum size of the string: ");

scanf("%d", &size);

str = (char \*)malloc((size + 1) \* sizeof(char));

if (str == NULL) {

printf("Memory allocation failed\n");

return 1;

}

printf("Enter the string: ");

scanf(" ");

fgets(str, size + 1, stdin);

int length = stringLength(str);

printf("Length of the string: %d\n", length);

free(str);

return 0;

}

Output:  
Enter the maximum size of the string: 30

Enter the string: siddhi nyati

Length of the string: 13  
  
b)Converting characters of string into upper case  
code:  
#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

void toUpperCase(char \*str) {

while (\*str) {

\*str = toupper(\*str);

str++;

}

}

int main() {

char \*str;

int size;

printf("Enter the maximum size of the string: ");

scanf("%d", &size);

str = (char \*)malloc((size + 1) \* sizeof(char));

if (str == NULL) {

printf("Memory allocation failed\n");

return 1;

}

printf("Enter the string: ");

scanf(" ");

scanf("%[^\n]", str);

toUpperCase(str);

printf("Uppercase string: %s\n", str);

free(str);

return 0;

}  
  
output:  
Enter the maximum size of the string: 30

Enter the string: siddhi

Uppercase string: SIDDHI  
  
c) Concatenating two strings to form a new string  
code: