**King Fahd University of Petroleum and Minerals**

**Information and Computer Science Department**

ICS 103: Computer Programming in C

**Summer Semester 2010-2011 (Term-103)**

##### Final Exam

**Time: 150 minutes (2 hours 30 minutes) Wednesday August 17, 2011**

|  |  |
| --- | --- |
| **Name:** |  |
|  **ID#:** |  |  |  |  |  |  |  |  |  |  |  |

PLEASE CIRCLE YOUR SECTION BELOW:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Section | 01 | 02 | 03 | 04 | 05 |
| Instructor | Dr. Aiman | Dr. Aiman | Mr. Faisal | Mr. Putu | Dr. Azzedin |

|  |  |  |
| --- | --- | --- |
| Question # | Maximum Marks | Obtained Marks |
| 1 | 36 |  |
| 2 | 20 |  |
| 3 | 20 |  |
| 4 | 24 |  |
| Total | 100 |  |

 **Notes.** 1. Make sure you have **15** pages including the cover page.

 2. Closed book and notes

 3. Write clearly, briefly and precisely

 4. Cheating will result in ZERO grade

##  Good Luck

## Question 1: (36 points)

**Determine the output of each of the following programs:**

|  |
| --- |
| **#include <stdio.h> // P1: 6 points** **#include <string.h>****#include <stdlib.h>****int check(char[], int, int, char);****int main() {** **char s[]="ABCDEFGHIJKLMNOPQRSTUVWXYZ";** **char target = 'C';** **int index = check(s, 0, strlen(s) - 1, target);** **printf("%d\n", index);** **system("Pause");** **return 0;****}****int check(char x[], int low, int high, char target) {** **int middle;**  **if (low > high) return -1;**  **middle = (3\*low + high)/4;**  **printf("New-mid = %c\n", x[middle]);** **if (x[middle] == target) return (middle);**  **else if (x[middle] < target)**  **return check(x, middle+1,high,target);** **else** **return check(x, low, middle-1,target);****}** |
| **Output:** |

|  |
| --- |
| **#include <stdio.h> // P2: 6 points****#include <stdlib.h>****int main() {** **int a[3][3];** **int i,j;** **for(i = 0; i < 3; i++) {** **for(j = i + 1; j < 3; j++)** **a[i][j] = i + j;** **}** **for(i = 0; i < 3; i++) {** **for(j = 0; j < i; j++)**  **a[i][j] = a[j][i];** **}** **for(i = 0; i < 3; i++)** **a[i][i] = 10\*(i + 1);** **for(i = 0; i < 3; i++) {** **for(j = 0; j < 3; j++)**  **printf("%d\t", a[i][j]);** **printf("\n");** **}** **system("Pause");** **return 0;****}** |
| **Output:** |

|  |
| --- |
| **#include <stdio.h> // P3: 6 points****#include <stdlib.h>****#include <ctype.h>****#include <string.h>****int main(void) {** **char pass[20];** **int i, valid = 1;** **printf("Type password> ");** **gets(pass);** **if(!isalpha(pass[0])) valid = 0;** **for(i = 1; i < strlen(pass) - 1; i++) {** **if(!isalnum(pass[i])) {**  **valid = 0;** **break;** **}** **if(i%2 == 0 && !isdigit(pass[i])) {** **valid = 0;** **break;** **}** **}** **if(!isupper(pass[i])) valid = 0;** **if(valid) printf("%s is valid\n", pass);** **else printf("%s is not valid\n", pass);** **system("Pause");** **return 0;****}**  |
| **What is the output for each of the following inputs:****Type password> oa9pJ****Type password> oa****Type password> SR104U** |

|  |
| --- |
| **#include <stdio.h> //P4: 6pts****#include <stdlib.h>****#include <string.h>****int main(void) {** **char input[80], output[80], oldtoken[80];** **char\* token;** **printf("Enter input string>\n");** **gets(input);** **token = strtok(input, " ");** **strcpy(oldtoken, token);** **strcpy(output, "");** **while(token != NULL) {** **if(strcmp(oldtoken, token) > 0) {** **strcat(output, oldtoken);** **}** **else if(strcmp(oldtoken, token) < 0) {** **strcat(output, token);** **}** **strcat(output, " ");** **strcpy(oldtoken, token);** **token = strtok(NULL, " ");** **}** **puts(output);** **system("Pause");** **return 0;****}** |
| **What is the output for each of the following inputs:****Enter input string>****alpha beta gamma** **Enter input string>****gamma beta alpha** |

|  |
| --- |
| **#include <stdio.h> //P5: 6pts****#include <stdlib.h>****#include <string.h>****int main(void) {** **char phrases[80][20], temp[80];** **int i = 0, j = 0, size;** **printf("Type phrases (type ‘end’ to terminate):\n");** **do {** **gets(phrases[i]);** **} while(strcmp(phrases[i++], "end") != 0);** **size = i;** **for(i = 0; i < size - 2; i++)**  **for(j = i + 1; j < size - 1; j++)** **if(strlen(phrases[i]) > strlen(phrases[j]))**  **{** **strcpy(temp, phrases[i]);** **strcpy(phrases[i], phrases[j]);** **strcpy(phrases[j], temp);** **}** **printf("\nThe output is:\n");**  **for(i = 0; i < size - 1; i++)** **puts(phrases[i]);** **system("Pause");** **return 0;****}** |
| **What is the output if the user input is:****Type phrases (end to terminate):****kfupm****ics 103****grades****final exam****end** |
| **#include <stdio.h> //P6 6pts****#include <stdlib.h>****void test(int, int\*);****int main(void) {** **int val1 = 45, val2 = 37;** **int flag1, flag2;** **test(val1, &flag1);** **test(val2, &flag2);** **if(flag1)**  **printf("%d passed test\n", val1);** **if(flag2)**  **printf("%d passed test\n", val2);** **system("Pause");** **return 0;****}****void test(int val, int\* result) {** **int rem, sum = 0;** **do {** **rem = val % 10;** **val = val/10;** **if(rem % 3 == 0) continue;** **sum = sum + rem;** **} while(val != 0);** **printf("Sum = %d\n", sum);** **if(sum % 3 == 0) \*result = 1;** **else \*result = 0;****}** |
| Output: |

**Question 2: ( 20 points )**

You are required to write a C program to read a two dimensional array of integers, exchange two rows, exchange two columns and display the array after being read and after each exchange. In your solution, you are required to do the following:

1. Ask the user to enter the number of rows and columns in the array and read them. Assume that the maximum array size is 10x10.
2. Write a function to read a two-dimensional array and use it to read the array.
3. Write a function to display a two-dimensional array and use it to display the read array.
4. Write a function to exchange two rows in the array.
5. Write a function to exchange two columns in the array.
6. Display a menu for the user asking him to select a choice between exchanging two rows, exchanging two columns or exit the program. The menu should be displayed repeatedly until the user chooses to exit the program. If an incorrect choice is entered, the message “Invalid choice !!” should be displayed.

In all functions above, the **dimensions** of the array must be passed as parameters.

*A sample execution of the program is shown below:*



**#include <stdio.h>**

**#include <stdlib.h>**

**#define size 10**

**void menu(){**

 **printf("Select one of the choices below:\n");**

 **printf("1. Exchange two rows\n");**

 **printf("2. Exchange two columns\n");**

 **printf("3. Exit program \n");**

**}**

**void rarray(int a[][size],int n, int m){**

 **printf("Enter %dx%d array of inetegers:\n",n,m);**

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

 **for (int j=0;j<m;j++)**

 **scanf("%d",&a[i][j]);**

**}**

**void darray(int a[][size],int n, int m){**

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

 **for (int j=0;j<m;j++)**

 **printf("%3d ",a[i][j]);**

 **printf("\n");**

 **}**

**}**

**void xchgr(int a[][size], int m, int r1, int r2){**

 **int t;**

 **for (int j=0;j<m;j++){**

 **t=a[r1][j];**

 **a[r1][j]=a[r2][j];**

 **a[r2][j]=t;**

 **}**

**}**

**void xchgc(int a[][size], int n, int c1, int c2){**

 **int t;**

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

 **t=a[i][c1];**

 **a[i][c1]=a[i][c2];**

 **a[i][c2]=t;**

 **}**

**}**

**int main(void) {**

 **int a[size][size];**

 **int row, col, v1, v2, choice;**

 **printf("Enter the number of rows and columns: ");**

 **scanf("%d %d",&row, &col);**

 **rarray(a,row,col);**

 **printf("You have entered the following %dx%d array:\n", row, col);**

 **darray(a,row,col);**

 **menu();**

 **scanf("%d",&choice);**

 **while (choice !=3){**

 **switch(choice){**

 **case 1:**

 **printf("Enter the two rows to exchange: ");**

 **scanf("%d %d",&v1,&v2);**

 **xchgr(a,col,v1,v2);**

 **printf("Array after exchanging row %d & row %d:\n",v1, v2);**

 **darray(a,row,col); break;**

 **case 2:**

 **printf("Enter the two columns to exchange: ");**

 **scanf("%d %d",&v1,&v2);**

 **xchgc(a,row,v1,v2);**

 **printf("Array after exchanging column %d & column %d:\n",v1, v2);**

 **darray(a,row,col); break;**

 **default:**

 **printf("Invalid choice!!\n");**

 **}**

 **menu();**

 **scanf("%d",&choice);**

 **}**

 **system("pause");**

 **return 0;**

**}**

**Question 3: ( 10 + 10 = 20 points )**

Write a solution in C language for the following:

1. [**10 points**] The function **insertStr** to insert a string into another string at the specified index/location. The function must have the prototype

void insertStr(char destination[], char source[], int start);

where the string **source** will be inserted into the string **destination** at the position **start**.

It assumes that the destination string has big enough size. If the position **start** is out of range the function will do nothing. For example, given a **destination** of "*An Introduction Course*" and **source** of "*to C Programming*" and **start** of 16, the function should transform **destination** to "*An Introduction to C Programming Course*".

1. [**10 points**] The function **removeChars** that deletes characters from a string. The function must have the prototype

void removeChars(char str[], char remove[]);

where any character existing in **remove** must be deleted from **str**.

For example, given a **str** of "*Battle of the Vowels:Hawaii*" and **remove** of "*aeiou*", the function should transform **str** to "*Bttl f th Vwls:Hw*".

*Hint*: You may write additional helper functions to solve the problem.

v**oid insertStr(char destination[], char source[], int start);**

**void removeChars(char str[], char remove[]);**

**int findChr(char str[], char a);**

**void insertStr(char destination[], char source[], int start)**

**{**

 **int destSize = strlen(destination);**

 **int srcSize = strlen(source);**

 **//check if start is in the range**

 **if (start > destSize) return;**

 **//Move the characters from start to start+strlen(source)**

 **//The move must be repeated in backward direction starting from the last character**

 **for(int i=destSize-1; i>=start; i--)**

 **destination[i+srcSize] = destination[i];**

 **//Copying the source into destination**

 **for(int j=0; j<srcSize; j++)**

 **destination[start + j] = source[j];**

 **destination[destSize + srcSize] = '\0';**

 **return;**

**}**

**void removeChars(char str[], char remove[])**

**{**

 **int sizeStr = strlen(str);**

 **int k;**

 **for(int i=sizeStr-1; i>=0; i--)**

 **{**

 **if (findChr(remove, str[i]) != -1)**

 **//copying the remaining characters**

 **for(k=i+1; k<=strlen(str); k++)**

 **str[k-1] = str[k];**

 **}**

**}**

**int findChr(char str[], char a)**

**{**

 **for(int i=0; str[i]!='\0'; i++)**

 **if (str[i]==a) return i;**

 **return -1;**

**}**

**Question 4: ( 24 points )**

Write a C program that does the following:

1. Reads a list of names and scores from the file “input.txt” and stores them in appropriate arrays. Assume that the name is composed of initial and family name. Assume that your program does not know the size of data and that the maximum number of names and scores is 100 and the maximum name length is 80.
2. The **bubble sort algorithm** to sort an array of doubles in decreasing order is given below as a function. Modify the function and use it to sort the data read in part (i) in **increasing alphabetical order** and display the sorted data on the screen.

**void bubble\_sort(double a[], int size) {**

 **int i, pass = 1, swap\_occurs, t;**

 **do{**

 **swap\_occurs = 0;**

 **for(i = 1; i <= size - pass; i++)**

 **if (a[i-1] < a[i]) {**

 **t=a[i-1]; a[i-1]=a[i]; a[i]=t;**

 **swap\_occurs = 1;**

 **}**

 **pass++;**

 **} while (swap\_occurs && pass <= size-1);**

**}**

1. A recursive implementation of the **binary search algorithm** is given in the **binary\_search** function below. Modify the implementation of the function to a **non-recursive** implementation for searching names instead of searching scores.

**int binary\_search (double x[], int low, int high, double target) {**

 **int middle;**

 **if (low > high) return -1;**

 **middle = (low + high)/2;**

 **if (x[middle] == target) return (middle);**

 **else if (x[middle] < target)**

 **return binary\_search(x, middle+1,high,target);**

 **else**

 **return binary\_search(x, low, middle-1,target);**

**}**

1. Ask the user to enter his name and then using your implementation of the **binary\_search** function in (iii), print the student name along with his score. If the student is not found, the program should print that the name is not found.

*Sample execution of the program is given below:*

|  |  |
| --- | --- |
| **input.txt** | **Output** |
| Sami Jaber 90.5Ali Saleh 85.0Ahmad Amin 100.0Saud Muhammad 70.5Khaled Ibrahim 65.5 | Ahmad Amin 100.00Ali Saleh 85.00Khaled Ibrahim 65.50Sami Jaber 90.50Saud Muhammad 70.50Enter the student name: Sami JaberThe student Sami Jaber has a score of 90.50 |

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#define SIZE 100**

**void SortNames(char NAMES[][80], double SCORES[], int count);**

**int SearchName(char NAMES[][80], int low, int high, char name[]);**

**int main() {**

 **FILE \*infile;**

 **int status, count, index;**

 **double SCORES[SIZE];**

 **char NAMES[SIZE][80], first[80], last[80], full[80], name[80];**

 **infile = fopen("input.txt","r");**

 **count=0;**

 **status = fscanf(infile, "%s%s%lf", first, last, &SCORES[count]);**

 **while (status != EOF){**

 **strcpy(full, first);**

 **strcat(full, " ");**

 **strcat(full, last);**

 **strcpy(NAMES[count], full);**

 **count++;**

 **status = fscanf(infile, "%s%s%lf", first, last, &SCORES[count]);**

 **}**

 **fclose(infile);**

 **SortNames(NAMES, SCORES, count);**

 **for (int i=0; i< count; i++)**

 **printf("%s\t%.2f\n",NAMES[i],SCORES[i]);**

 **printf("\n");**

 **printf("Enter the student name: ");**

 **gets(name);**

 **index=SearchName(NAMES, 0, count-1, name);**

**if (index>=0)**

 **printf("The student %s has a score of %6.2f\n\n", name, SCORES[index]);**

 **else**

 **printf("The student %s is not found\n\n", name);**

 **system("pause");**

 **return 0;**

**}**

**void SortNames(char NAMES[][80], double SCORES[], int count){**

 **int i, pass = 1, swap\_occurs;**

 **char tempn[80];**

 **double temps;**

 **do{**

 **swap\_occurs = 0;**

 **for(i = 1; i <= count - pass; i++) {**

 **if (strcmp(NAMES[i-1], NAMES[i])>0) {**

 **temps=SCORES[i-1]; SCORES[i-1]=SCORES[i]; SCORES[i]=temps;**

 **strcpy(tempn, NAMES[i-1]);**

 **strcpy(NAMES[i-1], NAMES[i]);**

 **strcpy(NAMES[i], tempn);**

 **swap\_occurs = 1;**

 **}**

 **}**

 **pass++;**

 **} while (swap\_occurs && pass <= count-1);**

**}**

**int SearchName(char NAMES[][80], int low, int high, char name[]){**

 **int middle;**

 **while (low <= high) {**

 **middle = (low + high)/2;**

 **if (!strcmp(NAMES[middle],name)) return (middle);**

 **else if (strcmp(NAMES[middle],name)<0)**

 **low=middle+1;**

 **else**

 **high=middle-1;**

 **}**

 **return -1;**

**}**