2022 CAT-CERT 스터디 보고서

컴퓨터정보공학부 202221168 김지혜

2022.04.08 #C\_study-3

Calc 함수 완성!

소스코드

#include <stdio.h>

int Calc(int a, char symbol, int b);

int Calc(int a, char symbol, int b) {

int result;

if (symbol == '+' ) {

result = a + b;

}

else if (symbol == '-') {

result = a - b;

}

else if (symbol == '/') {

result = a / b;

}

else if (symbol == '\*') {

result = a \* b;

}

return result;

}

int main()

{

int a, b, answer;

char symbol;

scanf("%d %c %d", &a, &symbol, &b);

answer = Calc(a, symbol, b);

printf("%d\n", answer);

}

출력 결과

텍스트이(가) 표시된 사진

자동 생성된 설명

배열 실습

소스코드

#include<stdio.h>

int main()

{

int AV = 0;

int i;

int arr[5] ={0, };

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

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

AV += arr[i];

}

AV /= 5;

printf("AV : %d", AV);

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

#실습과제 1

소스코드

#include <stdio.h>

int main()

{

int answer, tmp;

int arr[3];

for (int i = 0; i < 3; i++) {

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

}

for (int i = 0; i < 3; i++) {

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

if (arr[j] < arr[j + 1]) {

tmp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = tmp;

}

}

}

int count = 0;

int differ1= 0;

int differ2 = 0;

int differ3 = 0;

int result;

int min=0;

differ1 = arr[0] - arr[1];//4 1

differ2 =arr[1] - arr[2]; //2 1

differ3 = arr[0] -arr[2]; //6 2

if (differ1 == (differ2 \* 2) || differ2 == (differ1 \* 2) || differ3 == (differ1 \* 2)) {

if (differ1 > differ2) {

min = differ2;

result = arr[1] + min;

}

else if (differ1 < differ2) {

min = differ1;

result = arr[1] - min;

}

else if (differ1 == differ2) {

min = differ1;

result = arr[2] - min;

}

}

printf("%hhd\n", result);

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

#실습과제 2

소스코드

#include <stdio.h>

int main() {

char str[100000];

int arr[26] = { 0, };

int i;

int j;

int num=0;

int max = 0;

int answer = 0;

int result = 0;

scanf("%s", &str);

for (j = 0; str[j] != '\x00'; j++) {

for (i = 65; i < 91; i++) {

if ((int)str[j] == i || (int)str[j]==i+32){

arr[i - 65] += 1;

}

}

}

for (i = 0; i < 26; i++) {

if (max < arr[i] && arr[i]!=0) {

max = arr[i];

answer = i + 65;

}

else if (max == arr[i] && arr[i] != 0) {

answer = '?';

}

}

if (max != arr[i]) {

printf("%c\n", answer);

}

else if (arr[i] == max) {

printf("%c\n",answer);

}

return 0;

}

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명출력결과

#실습과제 3

소스코드

#include<stdio.h>

#include<string.h>

#include<time.h>

int main() {

srand(time(NULL));

char wordle[70][6] = {"about","above","abuse", "alert", "align", "alive", "alone", "amaze", "angel", "ankle", "array", "azure", "build", "cause", "curse", "cycle", "daily", "diary", "earth", "email", "entry", "every", "extra", "faith", "field", "fusil", "guess", "happy", "house", "human", "image", "juice", "knock", "knife", "kitty", "limit", "lunch", "maybe", "media", "movie", "naïve", "noise", "ocean", "olive", "opera", "other", "piano", "pizza", "quiet", "radio", "right", "sauce", "sharp", "slice", "solid", "space", "stone", "their", "today", "under", "unity", "union", "value", "video", "voice", "which", "wrong", "young", "youth", "zebra"};

int num = rand() % 70; // 제시된 단어들 고르기위한 번호

int count = 1;

char str[6]; //내가 입력할 문자

printf("5글자 단어를 입력하시오: ");

scanf("%s", str);

while (count <= 6) {

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

if (wordle[num][i] == str[i])

printf("%c = green\n", str[i]);

else if ((wordle[num][i] != str[i]) && (strchr(wordle[num], str[i]) != NULL))

printf("%c = yellow\n", str[i]);

else

printf("%c = grey\n", str[i]);

}

if (strcmp(wordle[num], str) == 0) {

printf("축하합니다. %d 회만에 맞추셨습니다.\n", count);

break;

}

count += 1;

printf("5글자 단어를 입력하시오: ");

scanf("%s", str);

if (count == 6) {

printf("실패하셨습니다. 답은 %s입니다\n", wordle[num]);

break;

}

}

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

#실습과제 4

소스코드

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

int main() {

int x = 0;

int y = 0;

printf("x축의 크기 (2-9)?");

scanf("%d", &x);

printf("y축의 크기 (2-9)?");

scanf("%d", &y);

int i;

int j;

int a;

int b, k;

int arr[11][11];

int count = 0;

srand(time(NULL));

for (i = 0; i < y; i++) {

for (j = 0; j < x; j++) {

arr[i][j] = rand() % 2;

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

}

printf("\n");

}

while (1) {

printf("xy좌표(두자리수)는?(종료를 원하면 0을 누르세요) :");

scanf("%d", &k);

if (k== 0)

break;

a = k / 10;

b = k % 10;

if (a < x || b < y) { //기본

if (arr[y - b][a - 1] == 0) {

arr[y - b][a - 1] = 1;

}

else {

arr[y - b][a - 1] = 0;

}

if (arr[y - b][a] == 0) {

arr[y - b][a] = 1;

}

else {

arr[y - b][a] = 0;

}

if (arr[y - b][a - 2] == 0) {

arr[y - b][a - 2] = 1;

}

else {

arr[y - b][a - 2] = 0;

}

if (arr[y - 1 - b][a - 1] == 0) {

arr[y - 1 - b][a - 1] = 1;

}

else {

arr[y - 1 - b][a - 1] = 0;

}

if (arr[y - b + 1][a - 1] == 0) {

arr[y - b + 1][a - 1] = 1;

}

else {

arr[y - b + 1][a - 1] = 0;

}

}

if (a == x && b == y) { // 끝부분일 때

if (arr[y - b][a - 1] == 0) {

arr[y - b][a - 1] = 1;

}

else {

arr[y - b][a - 1] = 0;

}

if (arr[y - b][a - 2]==0) {

arr[y - b][a - 2] = 1;

}

else {

arr[y - b][a - 2] = 0;

}

if (arr[y - b +1][a - 1] == 0) {

arr[y - b +1][a - 1] = 1;

}

else {

arr[y - b +1][a - 1] = 0;

}

}

for (i = 0; i < y; i++) {

for (j = 0; j < x; j++) {

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

if (arr[i][j] == 0) {

count++;

}

}

printf("\n");

}

if (count == x \* y) {

printf("\n축하합니다\n");

break;

}

count = 0;

}

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

조사 과제

1.난수 관련 함수에 대해 조사 및 실습

(헤더가 무엇인지 간단하게)

난수를 생성하는 rand()함수와 srand()함수가 있음

Rand() – 난수의 생성패턴을 한 개로 설정

Srand()- 난수의 생성패턴을 여러 개로 설정

rand함수에 사용될 수를 초기화함. 초기화를 매개변수로 받는 seed값을 이용해서 함. rand함수는 내부적으로 어떤 srand의 매개변수로 들어온 seed값과 매칭되는 숫자가 정해짐. seed값에 의해서 rand함수의 결과값이 변할 수 있음

srand의 인자 값을 계속 바꾸어가며 실행하면 각 다른 난수들이 출력됨. 수동으로 바꾸기에는 벅차기 때문에 자동으로 바꾸는 방법이 있음 – 현재시간을 seed값으로 넣으면 됨

c언어에서 사용하는 랜덤함수(rand)를 사용하려면 <stdilb.h>헤더파일을 포함시켜야함

**헤더파일**

* 코드 맨 위에 존재하고 있으며, 개발자가 쉽게 코딩을 하도록 함수나 클래스를 미리 지정해놓은 파일. 컴퓨터는 스스로 생각하여 해당 단어가 의미하는 기능을 알지 못함. 직접 정의를 해주고 어떻게 작동하는 지 설정해줘야함

난수 함수 실습

소스코드

#include <stdio.h>

#include<stdlib.h>

#include<time.h>

void main()

{

srand((unsigned int)time(NULL));

printf("난수 : %d\n", rand());

printf("난수 : %d\n", rand());

printf("난수 : %d\n", rand());

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

2. call by reference & call by value 조사 및 구현

함수의 호출방법으로

Call by value(값에 의한 호출)

Call by reference(참조에 의한 호출)

함수 호출 : 정의된 함수를 호출하는 것. 함수에 정의한 매개변수의 형태에 따라 결정됨

|  |  |
| --- | --- |
| **Call by value(값에 의한 호출)** | **Call by reference(참조에 의한 호출)** |
| * 함수가 호출될 때 메모리 공간 안에서는 함수를 위한 별도의 임시공간이 생성됨(종료시 사라짐) * 함수 호출 시 전달되는 변수 값을 복사해서 함수 인자로 전달 * \*복사된 인자는 함수 안에서 지역적으로 사용됨 * 함수안에서 인자값이 변경되더라도 외부 변수 값은 변경이 안됨   장점 : 복사하여 처리하기 때문에 안전함. 원래의 값이 보존이 된다  단점: 복사를 하기 때문에 메모리 사용량이 늘어난다 | * 메모리 공간에서는 함수를 위한 별도의 임시공간이 생성됨 * 함수 호출 시 전달되는 변수의 참 조를 전달함(변수를 가르킴) * 함수안에서 인자값이 변경되면 아규먼트로 전달된 객체의 값도 변경됨   \*아규먼트 : 함수를 이용하는 사람이 매개변수에 집어 넣는 값  장점 : 복사하지 않고 직접 참조를 하기에 빠름  단점: 직접 참조를 하기에 원래 값이 영향을 받는다 |

소스코드

#include <stdio.h>

void func(int n) {

n = 20;

}

void main() {

int n = 10;

func(n);

printf("%d\n", n);

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

소스코드

#include <stdio.h>

void func(int \*n) {

\*n = 20;

}

void main() {

int n = 10;

func(&n);

printf("%d\n", n);

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

3. main 함수의 argc, argv에 대해 조사 및 실습

main함수는 프로그램에서 최초로 실행되는 곳.

main함수의 매개변수

* int argc-main()함수에 전달되는 데이터의 개수

- char\* argv[]-main()함수에 전달되는 실제적인 데이터로 char형 포인터 배열로 구성

\*argc의 개수가 조건에 만족하지 못하면 에러메세지 출력

소스코드

#include <stdio.h>

int main(int argc, char\* argv[])

{

int i = 0;

printf("전달된 인자 갯수: %d\n", argc);

for (i = 0; i < argc; i++)

{

printf("%d번째 문자열: %s\n", i + 1, argv[i]);

}

return 0;

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

4.메모리 영역(code, data, bss, stack, heap)에 대해 조사

|  |  |
| --- | --- |
| 코드(code) 영역 | 실행할 프로그램의 코드가 저장되는 영역  CPU는 코드 영역에 저장된 명령어를 하나씩 가져가서 처리함 |
| 데이터(data) 영역 | 프로그램의 전역 변수와 정적(static) 변수가 저장되는 영역  프로그램의 시작과 함께 할당되며, 프로그램이 종료되면 소멸  - **Initialized Data Segment**  프로그래머가 직접 초기화를 해준 전역변수와 정적변수가 저장됨  **Uninitialized Data Segment = BSS**  초기화 되지 않은 전역변수(global)와 정적변수(static)가 저장됨.  초기화 되지 않은 변수는 커널이 자동으로 0으로 초기화시킴. |
| 스택(stack)영역 | 함수의 호출과 관계되는 지역 변수와 매개변수가 저장되는 영역  함수의 호출과 함께 할당되며, 함수의 호출이 완료되면 소멸  푸시(push) 동작으로 데이터를 저장, 팝(pop) 동작으로 데이터를 인출  **후입선출 출력시 더미의 가장 위에 있는 데이터를 빼옴** |
| 힙(heap) 영역 | 사용자가 직접 관리할 수 있는 그리고 해야만 하는 메모리 영역  사용자에 의해 메모리 공간이 동적으로 할당되고 해제  메모리의 낮은 주소에서 높은 주소의 방향으로 할당 |

5. static변수에 대해 조사 및 실습

:정적변수

선언 방법 : static 자료형 변수이름;

함수내부에서, 함수 외부에서 사용 가능

static변수

직접 초기화해주지 않아도 자동으로 0으로 초기화

프로그램이 시작될 때 할당되고 프로그램이 끝날 때 파괴됨

함수가 실행될 때 생성되고, 함수가 끝날 때 파괴되는 지역변수와 달리 함수가 끝나도 그 값을 기억하고 있음

전역으로 선언한 static변수는 소스파일 내의 모든 함수에서 사용 가능

(다른 소스파일에서는 사용하지 못함 => 정보를 숨기는 효과O)

=외부 정적변수

함수 내부에서 선언하면 다른 함수에선 값을 참조할 수 없음

=내부 정적변수

\*일반변수로 선언하면 함수가 실행될 때마다 선언 및 초기화가 됨

소스코드

#include <stdio.h>

static int Number;

void Test1() {

Number += 10;

}

void Test2() {

Number += 20;

}

int main() {

printf("정적변수 Numebr의 값 : %d\n", Number);

Test1();

printf("Test1 실행 후 Number의 값 : %d\n", Number);

Test2();

printf("Test2 실행 후 Number의 값 : %d\n", Number);

return 0;

}

출력결과

텍스트이(가) 표시된 사진

자동 생성된 설명

6. Calling Convention: cdecl 와 stdcall 에 대해 조사

함수 호출 규약

-함수를 호출할 때 파라미터(매개변수)를 전달하는 방식에 대한 약속

Cdecl 방식

각 함수 호출에 스택정리 코드를 포함시킴

-오른쪽에서 왼쪽으로 매개이자를 스택프레임에 삽입

-함수 호출 시 넘어가는 매개인자의 개수가 일정하지 않아서 피호출자는 스택을 스스로 정리하기 어려워 호출자가 스택을 정리한 후 함수를 실행하고, 실행을 마친 후 다시 호출자가 자신이 삽입했던 인자들을 제거하면서 스택을 정리함

-함수 이름앞에 “\_”를 붙이지만 적용되는 함수가 가변인자함수여서 매개인자의 데이터 크기의 합을 알기 힘들어 “@”와 10진수의 데이터 크기부분은 표시하지 않음

**stdcall 방식**

-오른쪽에서 왼쪽으로 매개이자를 스택프레임에 삽입

인자의 개수가 고정적인Win32 API에서 사용되는 방식

함수 호출 시 넘어가는 매개인자의 고정적이기 때문에 스택 프레임에 삽입, 삭제되는 인자들이 항상 일정함. 그래서 함수 호출 시 피호출자는 스스로 스택을 정리한 후 함수를 실행하고, 실행을 마친 후 자신이 삽입했던 인자들을 제거하면서 스택을 정리하고 함수를 빠져나옴

1. 함수이름앞에 “\_”가 붙고 함수 이름뒤에는 “@”가 붙음
2. “@”뒤에는 10진 정수로 매개인자의 데이터 크기의 합을 표시함

**\*\_fastcall**

함수 호출 시 두개의 인자를 스택이 아닌 레지스터에 저장함

그리고 나머지 인자를 스택에 저장하는 방식

함수가 두개 이하의 인자를 사용하는 경우 스택에 인자를 저장하지 않아도 되므로 코드가 간결해지고 수행속도가 빠름

7.버블 정렬에 대해 조사

인접한 두 항목의 값을 비교해서 일정한 기준을 만족하면 서로의 값을 교환하여 정렬하는 방식

1. 구현이 매우 간단함
2. 하나의 요소가 가장왼쪽에서 오른쪽으로 이동하려면 배열에서 모든 다른요소들과 교환되어야함
3. 자료 교환작업(swap)이 자료의 이동작업보다 더 복잡해서 버블 정렬은 거의 쓰지 않음

+)재귀함수

자기자신을 호출하여 계속 불러오는 함수

함수는 반환이 되어야 끝이 나는데 재귀함수(Recursive() )를 만나면 함수는 끝나지 않고 계속 자신을 메모리 다른 곳에 복사 시켜 실행시킴. 마지막에 리턴이 되어야 종료됨

반복적인 호출이 가능한 이유는 메모리에 복사본을 만들기 때문임