Labor 3.1

#include <stdio.h>  
float sideLength;  
float dVolume;  
float dSurface;  
int restart;  
//Takes three inputs: a sidelength and two variables. Uses the input sidelength to define and store the Volume and Area  
//The other two variables point to and change the entered variables, so that their values are changed outside of the funct.  
float calculateCubeData(float dsideLength, float \*pdVolume, float \*pdSurface) {  
 if (dsideLength<0){ //Checks if sideLength is zero  
 printf("Invalid Input\n");  
 if (dsideLength<0){  
 printf("dsideLength must be more than 0\n");  
 }  
 return -1;  
 } else if (dsideLength > -1){  
 float Volume = dsideLength \* dsideLength \* dsideLength;  
 float Area = dsideLength \* dsideLength \* 6;  
 \*pdVolume = Volume;  
 \*pdSurface = Area;  
 return 0;  
 }  
}  
int main() {  
 start:  
 sideLength = 13323443; //If text is entered in the scanf() below, sidelength will not be changed and keep the value 13323443  
 restart =0;  
 printf("Please Enter Sidelength:\n");  
 scanf("%f", &sideLength);  
 if (sideLength == 13323443){ //If sidelength stays (which should only happen because of an invalid input), it will end print...  
 printf("Invalid Syntax\n");  
 } else {  
 printf("sidelength is: %f\n", sideLength);  
 calculateCubeData(sideLength, &dVolume, &dSurface);  
 if (sideLength > 0) {  
 printf("%f\n", dVolume);  
 printf("%f\n", dSurface);  
 }  
 printf("Again? 1=yes 2=no\n");  
 scanf("%d", &restart);  
 if (restart == 1) {  
 goto start;  
 }  
 }  
}

Info Labor 3.2:

#include <stdio.h>  
#include <string.h>  
#include <malloc.h>  
  
void concatStrings (char \*output, int inputLength, char \*input,...){  
 int i;  
 int charsInList=0;  
 for (i=0; i<(inputLength); i++){  
 charsInList += strlen((&input)[i]);  
 }  
 char \*list = malloc(charsInList\*sizeof(char));  
 strcpy(list, (&input)[0]);  
 int c;  
 for (c=1; c<inputLength; c++) {  
 strcat(list, (&input)[c]);  
 }  
 strcpy(output, list);  
}  
  
int main () {  
 char outputList[1000];  
 concatStrings(outputList,4,"Word","Second Word","\n","DDD");  
 int i;  
 for (i=0; i<strlen(outputList); i++) {  
 printf("%c ",outputList[i]);  
 }  
 return 0;  
}

Info Labor 3.3 Recursion:

#include <stdio.h>  
int fibonacciNum = 33234425;  
  
double fibonacciZahlen(int input) {  
 if (input>49 || input < 0) {  
 return -1;  
 } else {  
 int i;  
 int c=0;  
 int fibonacciNums[input];  
 fibonacciNums[0]=1;  
 fibonacciNums[1]=1;  
 for (i=2;i<(input+1);i++){  
 fibonacciNums[i] = (fibonacciNums[i-1]+fibonacciNums[i-2]);  
 c++;  
 }  
 return fibonacciNums[c+1];  
 }  
}  
  
int main() {  
 printf("Calculate Fibonacci Number a\_");  
 scanf("%d", &fibonacciNum);  
 printf("\n");  
 if (fibonacciNum==33234425 || fibonacciZahlen(fibonacciNum)<0){  
 printf("Invalid Input.\n");  
 } else {  
 printf("Fibonacci Number a\_%d is equal to: %g\n", fibonacciNum, fibonacciZahlen(fibonacciNum));  
 }  
 return 0;  
}

Info Labor 3.4:

#include <stdio.h>  
  
int switchWords (char \*input, char \*output) {  
 int index=0;  
 int indexOut=0;  
 int occurences=0;  
 while(input[index]!='\0'){  
 if ((input[index] == 'Z' || input[index] == 'z') && (input[index +1] == 'e') && (input[index +2] == 'h') && (input[index +3] == 'n')) {  
 output[indexOut] = 'S';  
 output[indexOut+1] = 'e';  
 output[indexOut+2] = 'c';  
 output[indexOut+3] = 'h';  
 output[indexOut+4] = 's';  
 index +=4;  
 indexOut +=5;  
 occurences++;  
 }  
 output[indexOut] = input[index];  
 index++;  
 indexOut++;  
 }  
 output[indexOut] = '\0';  
 if ((occurences>8)==1){  
 return -1;  
 }else{return 0;}  
}  
  
int main () {  
 char string[600];  
 char cleanStringO[700];  
 printf("Please enter your string:\n");  
 scanf("%[^'\n']",string);  
 switchWords(string,cleanStringO);  
 if (switchWords(string,cleanStringO)==0){  
 printf("%s\n", cleanStringO);  
 } else if (switchWords(string,cleanStringO)==-1){  
 printf("Too many occurences\n");  
 } else {  
 printf("ERROR");  
 }  
}

Info Labor 3.5 (Bad approach but works. See corrected version below):

#include <stdio.h>  
#include <string.h>  
  
char hexNum[20];  
  
long long int power(long long int num,long long int power) {  
 int i;  
 long long int numLLoc=num;  
 if (power >> 1) {  
 for (i = 0; i < (power-1); i++) {  
 numLLoc \*= num;  
 }  
 } else if (power == 0) {  
 numLLoc = 1LL;  
 } else if (power ==1){  
 numLLoc = num;  
 } else {  
 return -1;  
 }  
 return numLLoc;  
}  
  
long long int hexToDec(char \*input) {  
 long long int output=0LL;  
 long long int list[1000];  
 int i;  
 long long int c=0;  
 // printf("input=%s\n",input);  
 for (i = strlen(input); i-- >0;) {  
 // printf("input[%d]=%d\n",i, input[i]);  
 if (input[i] == 'A') {  
 list[c] = 10LL\*power(16LL,c);  
 //printf("Contains A\n");  
 } else if (input[i] == 'B') {  
 list[c] = 11LL\*power(16LL,c);  
 //printf("Contains B\n");  
 }else if (input[i] == 'C') {  
 list[c] = 12LL\*power(16LL,c);  
 //printf("Contains C\n");  
 }else if (input[i] == 'D') {  
 list[c] = 13LL\*power(16LL,c);  
 //printf("Contains D\n");  
 }else if (input[i] == 'E') {  
 list[c] = 14LL\*power(16LL,c);  
 //printf("Contains E\n");  
 }else if (input[i] == 'F') {  
 list[c] = 15LL\*power(16LL,c);  
 printf("list[%lld] = 15\*power(16,%lld) which = %lld\n",c, c, (15LL\*power(16LL,c)));  
 printf("power(16,%lld)= %lld\n",c, power(16LL,c));  
 } else {  
 list[c] = ((input[i])-48LL)\*power(16LL,c);  
 //printf("%d\n", (((input[i])-48)\*power(16,c)));  
 //printf("pow(16,%d)=%d\n",d, pow(16,d));  
 }  
 c++;  
 }  
 for (i = 0; i < strlen(input); i++) {  
 printf("list[%d] = %lld\n", i, list[i]);  
 output += list[i];  
 }  
 return output;  
}  
int main() {  
 // printf("power 15^7 = %d\n",power(15,7));  
 printf("Enter Hexadecimal\n");  
 scanf("%s", hexNum);  
 printf("Decimal Vers is: %lld\n", hexToDec(hexNum));  
 return 0;  
}

Labor 3.5 Corrected Version:

#include <stdio.h>  
#include <string.h>  
  
char hexNum[20];  
  
long long int power(long long int num,long long int power) {  
 int i;  
 long long int numLLoc=num;  
 if (power >> 1) {  
 for (i = 0; i < (power-1); i++) {  
 numLLoc \*= num;  
 }  
 } else if (power == 0) {  
 numLLoc = 1LL;  
 } else if (power ==1){  
 numLLoc = num;  
 } else {  
 return -1;  
 }  
 return numLLoc;  
}  
  
long long int hexToDec(char \*input){  
 if ((input[0]=='0')&&(input[1]=='x')){  
 printf("Input being read\n");  
 int i;  
 long long int output=0;  
 long long int list[1000];  
 long long int c = 0;  
 for (i = strlen(input); i-- >2;) {  
 if ((input[i]>47)&& (input[i]<58)){ //Decimals 0-9 entered as char are stored as decimal + 48  
 list[c] = ((input[i])-48LL)\*power(16LL,c);  
 }  
 if (((input[i])>64)&&(input[i]<71)) {  
 list[c] = ((input[i])-55LL)\*power(16LL,c);  
 }  
 c++;  
 }  
 for (i = 0; i < strlen(input); i++) {  
 printf("list[%d] = %lld\n", i, list[i]);  
 output += list[i];  
 }  
 return output;  
  
 } else {  
 printf("Input could not be read. Check syntax\n");  
 return -1;  
 }  
}  
int main() {  
 printf("Enter Hexadecimal\n");  
 scanf("%s", hexNum);  
 printf("Decimal Vers is: %lld\n", hexToDec(hexNum));  
 return 0;  
  
}