\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report: HW7

Author: F14076083 魏湧致 <zxc33991177@gmail.com>

Class: 資訊111 (乙班)

Description:

這題和hw4一樣是要在float,double與其bit pattern互換，但要利用struct和union來完成。

Union :

將float和int以及由23、8、1位元組成的struct union在一起，double則是和long long以及由52、11、1位元組成的struct union在一起。將數字轉為bit pattern的方法和hw4相同，將bit pattern轉為數字則是利用pow函式將bit pattern分別加到m,e,s中，最後在print出數字。

Struct:

數字轉為bit pattern - 將float和bit0到bit31放在同個struct，再用hw4的方法一個一個求出並print出，double則是和bit0到bit63放在同個struct，再用hw4的方法一個一個求出並print出來。

bit pattern轉為數字 – 直接將struct內的bit pattern m,e,s做運算得到結果。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Code:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

typedef union{

float f;

int bit;

struct{

int m : 23;

int e : 8;

int s : 1;

}pattern;

}flo\_union;

typedef union{

double d;

long long bit;

struct{

long long m : 52;

long long e : 11;

long long s : 1;

}pattern;

}dou\_union;

typedef struct{

float f;

struct{

int bit0 : 1; int bit1 : 1; int bit2 : 1; int bit3 : 1;

int bit4 : 1; int bit5 : 1; int bit6 : 1; int bit7 : 1;

int bit8 : 1; int bit9 : 1; int bit10 : 1; int bit11 : 1;

int bit12 : 1; int bit13 : 1; int bit14 : 1; int bit15 : 1;

int bit16 : 1; int bit17 : 1; int bit18 : 1; int bit19 : 1;

int bit20 : 1; int bit21 : 1; int bit22 : 1; int bit23 : 1;

int bit24 : 1; int bit25 : 1; int bit26 : 1; int bit27 : 1;

int bit28 : 1; int bit29 : 1; int bit30 : 1; int bit31 : 1;

}bit;

struct{

int m : 23;

int e : 8;

int s : 1;

}pattern;

}flo\_struct;

typedef struct{

double d;

struct{

int bit0 : 1; int bit1 : 1; int bit2 : 1; int bit3 : 1;

int bit4 : 1; int bit5 : 1; int bit6 : 1; int bit7 : 1;

int bit8 : 1; int bit9 : 1; int bit10 : 1; int bit11 : 1;

int bit12 : 1; int bit13 : 1; int bit14 : 1; int bit15 : 1;

int bit16 : 1; int bit17 : 1; int bit18 : 1; int bit19 : 1;

int bit20 : 1; int bit21 : 1; int bit22 : 1; int bit23 : 1;

int bit24 : 1; int bit25 : 1; int bit26 : 1; int bit27 : 1;

int bit28 : 1; int bit29 : 1; int bit30 : 1; int bit31 : 1;

int bit32 : 1; int bit33 : 1; int bit34 : 1; int bit35 : 1;

int bit36 : 1; int bit37 : 1; int bit38 : 1; int bit39 : 1;

int bit40 : 1; int bit41 : 1; int bit42 : 1; int bit43 : 1;

int bit44 : 1; int bit45 : 1; int bit46 : 1; int bit47 : 1;

int bit48 : 1; int bit49 : 1; int bit50 : 1; int bit51 : 1;

int bit52 : 1; int bit53 : 1; int bit54 : 1; int bit55 : 1;

int bit56 : 1; int bit57 : 1; int bit58 : 1; int bit59 : 1;

int bit60 : 1; int bit61 : 1; int bit62 : 1; int bit63 : 1;

}bit;

struct{

long long m : 52;

long long e : 11;

long long s : 1;

}pattern;

}dou\_struct;

int bit32(int Pt1, int i){

if(i==30 || i==22)

printf(" ");

if( (1<<i) & Pt1 ){

printf("1"); return (1<<i) & Pt1;

}

else{

printf("0"); return (1<<i) & Pt1;

}

}

int bit64(long long Pt2, int i){

unsigned long long m=1;

if(i==62 || i==51)

printf(" ");

if( (m<<i) & Pt2 ){

printf("1"); return (m<<i) & Pt2;

}

else{

printf("0"); return (m<<i) & Pt2;

}

}

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

{

int t=atoi(argv[1]);

flo\_union num1\_union; flo\_struct num1\_struct;

dou\_union num2\_union; dou\_struct num2\_struct;

if(t==1)

{

num1\_union.f=atof(argv[2]);

num1\_struct.f=atof(argv[2]); unsigned int \*Pt1=(unsigned int \*)&num1\_struct.f;

printf("union : ");

for (int i=31; i>=0; i--)

{

if( (1<<i) & num1\_union.bit )

printf("1");

else

printf("0");

if(i==31 || i==23)

printf(" ");

} printf("\n");

printf("struct : ");

num1\_struct.bit.bit0 =bit32(\*Pt1,31); num1\_struct.bit.bit1 =bit32(\*Pt1,30); num1\_struct.bit.bit2 =bit32(\*Pt1,29);

num1\_struct.bit.bit3 =bit32(\*Pt1,28); num1\_struct.bit.bit4 =bit32(\*Pt1,27); num1\_struct.bit.bit5 =bit32(\*Pt1,26);

num1\_struct.bit.bit6 =bit32(\*Pt1,25); num1\_struct.bit.bit7 =bit32(\*Pt1,24); num1\_struct.bit.bit8 =bit32(\*Pt1,23);

num1\_struct.bit.bit9 =bit32(\*Pt1,22); num1\_struct.bit.bit10=bit32(\*Pt1,21); num1\_struct.bit.bit11=bit32(\*Pt1,20);

num1\_struct.bit.bit12=bit32(\*Pt1,19); num1\_struct.bit.bit13=bit32(\*Pt1,18); num1\_struct.bit.bit14=bit32(\*Pt1,17);

num1\_struct.bit.bit15=bit32(\*Pt1,16); num1\_struct.bit.bit16=bit32(\*Pt1,15); num1\_struct.bit.bit17=bit32(\*Pt1,14);

num1\_struct.bit.bit18=bit32(\*Pt1,13); num1\_struct.bit.bit19=bit32(\*Pt1,12); num1\_struct.bit.bit20=bit32(\*Pt1,11);

num1\_struct.bit.bit21=bit32(\*Pt1,10); num1\_struct.bit.bit22=bit32(\*Pt1,9) ; num1\_struct.bit.bit23=bit32(\*Pt1,8) ;

num1\_struct.bit.bit24=bit32(\*Pt1,7) ; num1\_struct.bit.bit25=bit32(\*Pt1,6) ; num1\_struct.bit.bit26=bit32(\*Pt1,5) ;

num1\_struct.bit.bit27=bit32(\*Pt1,4) ; num1\_struct.bit.bit28=bit32(\*Pt1,3) ; num1\_struct.bit.bit29=bit32(\*Pt1,2) ;

num1\_struct.bit.bit30=bit32(\*Pt1,1) ; num1\_struct.bit.bit31=bit32(\*Pt1,0) ; printf("\n");

}

else if(t==2)

{

num2\_union.d=atof(argv[2]) ; long long m=1;

num2\_struct.d=atof(argv[2]); long long \*Pt2=(long long \*)&num2\_struct.d;

printf("union : ");

for (int i=63; i>=0; i--)

{

if( (m<<i) & num2\_union.bit )

printf("1");

else

printf("0");

if(i==63 || i==52)

printf(" ");

} printf("\n");

printf("struct : ");

num2\_struct.bit.bit0 =bit64(\*Pt2,63); num2\_struct.bit.bit1 =bit64(\*Pt2,62); num2\_struct.bit.bit2 =bit64(\*Pt2,61);

num2\_struct.bit.bit3 =bit64(\*Pt2,60); num2\_struct.bit.bit4 =bit64(\*Pt2,59); num2\_struct.bit.bit5 =bit64(\*Pt2,58);

num2\_struct.bit.bit6 =bit64(\*Pt2,57); num2\_struct.bit.bit7 =bit64(\*Pt2,56); num2\_struct.bit.bit8 =bit64(\*Pt2,55);

num2\_struct.bit.bit9 =bit64(\*Pt2,54); num2\_struct.bit.bit10=bit64(\*Pt2,53); num2\_struct.bit.bit11=bit64(\*Pt2,52);

num2\_struct.bit.bit12=bit64(\*Pt2,51); num2\_struct.bit.bit13=bit64(\*Pt2,50); num2\_struct.bit.bit14=bit64(\*Pt2,49);

num2\_struct.bit.bit15=bit64(\*Pt2,48); num2\_struct.bit.bit16=bit64(\*Pt2,47); num2\_struct.bit.bit17=bit64(\*Pt2,46);

num2\_struct.bit.bit18=bit64(\*Pt2,45); num2\_struct.bit.bit19=bit64(\*Pt2,44); num2\_struct.bit.bit20=bit64(\*Pt2,43);

num2\_struct.bit.bit21=bit64(\*Pt2,42); num2\_struct.bit.bit22=bit64(\*Pt2,41); num2\_struct.bit.bit23=bit64(\*Pt2,40);

num2\_struct.bit.bit24=bit64(\*Pt2,39); num2\_struct.bit.bit25=bit64(\*Pt2,38); num2\_struct.bit.bit26=bit64(\*Pt2,37);

num2\_struct.bit.bit27=bit64(\*Pt2,36); num2\_struct.bit.bit28=bit64(\*Pt2,35); num2\_struct.bit.bit29=bit64(\*Pt2,34);

num2\_struct.bit.bit30=bit64(\*Pt2,33); num2\_struct.bit.bit31=bit64(\*Pt2,32); num2\_struct.bit.bit32=bit64(\*Pt2,31);

num2\_struct.bit.bit33=bit64(\*Pt2,30); num2\_struct.bit.bit34=bit64(\*Pt2,29); num2\_struct.bit.bit35=bit64(\*Pt2,28);

num2\_struct.bit.bit36=bit64(\*Pt2,27); num2\_struct.bit.bit37=bit64(\*Pt2,26); num2\_struct.bit.bit38=bit64(\*Pt2,25);

num2\_struct.bit.bit39=bit64(\*Pt2,24); num2\_struct.bit.bit40=bit64(\*Pt2,23); num2\_struct.bit.bit41=bit64(\*Pt2,22);

num2\_struct.bit.bit42=bit64(\*Pt2,21); num2\_struct.bit.bit43=bit64(\*Pt2,20); num2\_struct.bit.bit44=bit64(\*Pt2,19);

num2\_struct.bit.bit45=bit64(\*Pt2,18); num2\_struct.bit.bit46=bit64(\*Pt2,17); num2\_struct.bit.bit47=bit64(\*Pt2,16);

num2\_struct.bit.bit48=bit64(\*Pt2,15); num2\_struct.bit.bit49=bit64(\*Pt2,14); num2\_struct.bit.bit50=bit64(\*Pt2,13);

num2\_struct.bit.bit51=bit64(\*Pt2,12); num2\_struct.bit.bit52=bit64(\*Pt2,11); num2\_struct.bit.bit53=bit64(\*Pt2,10);

num2\_struct.bit.bit54=bit64(\*Pt2,9) ; num2\_struct.bit.bit55=bit64(\*Pt2,8) ; num2\_struct.bit.bit56=bit64(\*Pt2,7) ;

num2\_struct.bit.bit57=bit64(\*Pt2,6) ; num2\_struct.bit.bit58=bit64(\*Pt2,5) ; num2\_struct.bit.bit59=bit64(\*Pt2,4) ;

num2\_struct.bit.bit60=bit64(\*Pt2,3) ; num2\_struct.bit.bit61=bit64(\*Pt2,2) ; num2\_struct.bit.bit62=bit64(\*Pt2,1) ;

num2\_struct.bit.bit63=bit64(\*Pt2,0) ; printf("\n");

}

else if(t==3)

{

num1\_union.pattern.s=argv[2][0]-'0';

num1\_union.pattern.e=0;

num1\_union.pattern.m=0;

for (int i=0; i<8; i++)

num1\_union.pattern.e = num1\_union.pattern.e + (argv[3][i]-'0') \* pow(2,7-i);

for (int i=0; i<23; i++)

num1\_union.pattern.m = num1\_union.pattern.m + (argv[4][i]-'0') \* pow(2,22-i);

printf("union : %f\n",num1\_union.f);

num1\_struct.pattern.s=argv[2][0]-'0';

num1\_struct.pattern.e=-127;

num1\_struct.pattern.m=0;

for (int i=0; i<8; i++)

num1\_struct.pattern.e = num1\_struct.pattern.e + (int) ((argv[3][i]-'0') \* pow(2,7-i));

num1\_struct.f=pow(2,num1\_struct.pattern.e);

for (int i=0; i<23; i++)

num1\_struct.f = num1\_struct.f + (argv[4][i]-'0') \* pow(2,num1\_struct.pattern.e-i-1);

if(num1\_struct.pattern.s==-1)

num1\_struct.f=-num1\_struct.f;

printf("struct : %f\n",num1\_struct.f);

}

else if(t==4)

{

num2\_union.pattern.s=argv[2][0]-'0';

num2\_union.pattern.e=0;

num2\_union.pattern.m=0;

for (int i=0; i<11; i++)

num2\_union.pattern.e = num2\_union.pattern.e + (argv[3][i]-'0') \* pow(2,10-i);

for (int i=0; i<52; i++)

num2\_union.pattern.m = num2\_union.pattern.m + (argv[4][i]-'0') \* pow(2,51-i);

printf("union : %f\n",num2\_union.d);

num2\_struct.pattern.s=argv[2][0]-'0';

num2\_struct.pattern.e=-1023;

num2\_struct.pattern.m=0;

for (int i=0; i<11; i++)

num2\_struct.pattern.e = num2\_struct.pattern.e + (int) ((argv[3][i]-'0') \* pow(2,10-i));

num2\_struct.d=pow(2,num2\_struct.pattern.e);

for (int i=0; i<52; i++)

num2\_struct.d = num2\_struct.d + (argv[4][i]-'0') \* pow(2,num2\_struct.pattern.e-i-1);

if(num2\_struct.pattern.s==-1)

num2\_struct.d=-num2\_struct.d;

printf("struct : %f\n",num2\_struct.d);

}

return 0;

}

Compilation:

gcc -lm -o hw7 hw7.c

Execution:

./hw7 1 85.125

./hw7 2 85.125

./hw7 3 0 10000101 01010100100000000000000

./hw7 4 0 10000000101 0101010010000000000000000000000000000000000000000000

Output:

union : 0 10000101 01010100100000000000000

struct : 0 10000101 01010100100000000000000

union : 0 10000000101 0101010010000000000000000000000000000000000000000000

struct : 0 10000000101 0101010010000000000000000000000000000000000000000000

union : 85.125000

struct : 85.125000

union : 85.125000

struct : 85.125000