警告:このファイルを、俺の意に反して広めないこと。俺の意とは、アンケートに協力してくださった方だけに感謝の気持ちでファイルを公開したいから、それ以外の人に見せるのはしばらくしてからにしたい、というものである。

また、内容については責任を取らない(特に左辺値と右辺値)。

# 1.講義スライドの「C言語のポイント」を参考にして作成した問題

問題1

穴埋めせよ。①と②を答えてしまえば、後は思ったより少ない。

左辺値は＿①＿と考えてよい。一方、右辺値は＿②＿である。

(左辺値)=(右辺値);

は、

「(左辺値)」という名の＿①＿に、「(右辺値)」と書かれた＿②＿を入れるようなものである。

x=5;

は

＿①＿xに＿②＿5を入れる操作である。

x=5;

y=x;

は、

＿①＿xに＿②＿5をいれ、それから＿①＿yに＿①＿x**の中に入ってる＿②＿5**を入れる操作である。

今太字で示したが、＿①＿yに＿①＿xを入れる訳でない。

これは、「評価」を「紙を見たり、計算したり、＿③＿してくること」と考え、代入を「＿②＿を＿①＿に入れること」と考えればよいだろう[1]。

したがって、

(左辺値)=(左辺値);

という式もありえる。

しかし、

5=x;

のようなものは、＿②＿5に＿①＿xの中の＿②＿を入れろという無茶になるため、あり得ない。つまり左辺値(＿①＿)は、＿①＿から＿②＿を引っ張り出すことで右辺値(＿②＿)になることができるが、右辺値(＿②＿)は左辺値(＿①＿)にはなれないのである。

x=y=6は、=が＿④＿の演算子といって、＿⑤＿側を優先したがる性質があるので、

x=(y=6)のように扱われる。

＿⑥＿る操作であると考えてよいだろう。

C言語では、評価の結果は右辺値とすることになっている。つまり、＿①＿に＿②＿を入れた結果は、＿⑦＿であるということだ。よって、＿①＿xにも＿②＿6が入る。しかし、(x=y)=6は、＿⑧＿ることになり、

＿②＿に＿②＿6を入れるという無理な要求になってしまう。よって不適切である。

\*ptrというのは、ptrという＿⑩＿るものである。つまり、普通の＿①＿と見なせる。よって、

\*ptr=7というのは可能である。

問題2

++i++が何故ダメなのか、「副作用」を説明しつつ、評価、副作用、定まらないから という言葉を用いて言え。

問題3

副作用を起こす演算子の例を挙げよ。

問題4

10<9<8の真偽を説明せよ。

問題5

for(①;②;④)

{

③

}

をwhile文で書き換えよ。また、while文に書き換えられない条件を説明せよ。

問題6

while(①)

{

②

}

をfor文で書き換えよ。

問題7

配列は0-オリジンである。どういう意味か。

問題8

要素n個の配列の各要素について扱うfor文を、昇順のものと降順のもの1つずつかけ。

答え1

① 箱,② 紙,③ 箱から紙を引っ張り出,④ 右結合,⑤ 右,⑥ 箱xに、「箱yに紙6を入れたときの結果」を入れ,

⑦ 箱,⑧ 「箱xに箱ｙの中の紙を入れた結果」に紙6を入れ,⑨ 箱,⑩ ポインタを変数に変え

答え2

評価によって値が変わることを副作用という。++iやi++だけでは副作用が発生せず、++i;やi++;として初めて発生する。従って、++i++では副作用が発生する前に別の副作用がまた発生しようとしてしまうため、副作用が最終的にどうなるかが定まらないから

答え3

+=,-=,/=,\*=,?=,++,--

答え4

偽は0、真は非0の代表値として1であらわされるので、

10<9<8

偽<8

0<8

真

答え5

①

while(②)

{

③

④

}

但し、③が「continue;」だった場合、for文では④が実行され、while文では実行されないので、同じことにはならない。

答え6

for(;①;)

{

②

}

答え7

0から数え始める。

答え8

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

{

}

と

for(i=n-1;i>=0;i--)

{

}

[1] <https://www.ibm.com/support/knowledgecenter/ja/ssw_ibm_i_72/rzarg/lvalue.htm> によれば、

*すべての代入演算子は、それらの右方オペランドを評価し、その値を左方オペランドに代入します*

とのこと。

# 2.各講義スライドを参考に作成した問題

## 第一回講義

問題9

gccで、raw.cをapp.exeという名前でコンパイルしたいとき、どう打てばよいか。また、a.exeという名前でコンパイルしたいときはどうするか。

問題10

if文、for文などで出てくる{　～　}は何か。

問題11

{　～　}を用いないif文はどんな振る舞いをするか。

問題12

‘¥x41’がA,’\x61’がaの時、文字コードのみを用いて「Clang」と表示させるprintfをかけ。

問題13

穴埋めせよ。

int x=10;

printf("10進数＿①＿を８進で表すと＿②＿, 16進で表すと%＿③＿であ る。¥n",x,x,x);

問題14

問題13で、10進数を8進数、16進数に変換したわけだが、8進数や16進数の10を変換したいときには、  
int x=何 にすればよいか。

問題15

変数の初期化とは何か

問題16

問題1でも取り上げたとおり、変数=変数とすると、左辺にある変数はそのまま扱われるのに対して、

右辺にある変数は、その中にある右辺値を取り出すのに使われる。  
このように、場所によって変数に対する振る舞いが異なるが、これと関連してc言語以降の代入にまつわる不可解な点を「振る舞いが異なる」という言葉を用いて説明せよ。

答え9

gcc -o app raw.c

但し、a.exeという名前でよいときは

gcc raw.c

でよい。

答え10

ブロック。**ブロックも文の一種**だが、例外的に最後に「;」がつかない。複数の文をまとめるのに使う。

答え11

if()の直後の文一つだけが、if内の条件式が正しいときに実行される。他はすべて必ず実行される。

答え12

printf(“%c%c%c%c%c”, ‘\x43’, **’\x6c’**, ’\x61’, ‘\x64’, ’\x67’);

\x42→B,\x43→C…となることを類推したい。但し、16進数であることに気をつけよ(太字部)。

答え13

① %d,② %o,③ %x

それぞれDecimal, Octal, heXadecimalの意味だと思われる。

答え14

8進数→010

16進数→0x10

答え15

int i=5;

など、変数宣言と同時に値を決めること。**代入とは異なる**。

答え16

右辺が変数の場合と右辺が定数の場合で、コンパイラの振る舞いが異なる。

## 第二回講義

問題17

2項演算子の例を挙げよ

問題18

単項演算子の例を挙げよ

問題19

式とは

問題20

「x=3」に値はあるか(「x」でも「３」でもなく「x=3」にである)。あるならその値をいえ

問題21

num=2;

num=num+1;

とあるとき、

式num=num+1;

の評価の様子を答えよ。

問題22

aとbが実数型の時、a%bをいえ

問題23

穴埋めせよ。但し、a%bまたはa/bのどちらかで埋めること。

a = ＿①＿×b + ＿②＿

問題24

printfを使って、自身の成績がd評価かもしれないことについて、  
「俺さぼりまくったし、判定マジ100%d(不合格)だorz」といいたいときに生じる問題をいえ。また解決せよ。

問題25

++i

と

i++

の違いをいえ

問題26

a+=-bで、複合代入演算子はどれか。

問題27

式文とは何か。

問題28

演算子には優先順位がある。同じ優先順位の演算子が出てきたとき、どうするか。

a=b=cと、a+b-cの例を使って説明せよ。但し+と-の優先順位は同じである。

問題29

次の演算子を優先順位が高い順に並べよ。

++,&,\*,%,+,!,<.<=,==,!=,? :,=,+=,&&,||

問題30

int num=1;

としたとき、

num=num+1の値をいえ

また、

int inum;

double dnum=3.3;

としたとき、

inum=dnumの値をいえ。

問題31

キャスト演算子を用いて、xをdouble型の値に変換せよ

答え17

+,-,\*,/,=

答え18

正負を入れ替える「-」や、インクリメント、デクリメントの「++」,「--」

答え19

変数、定数そのもの または 変数、定数、式を演算子で結合したもの

答え20

ある。格納される値(なので右辺値)。

答え21

num=num+1

→ num=2+1

→ num=3

→ 3

答え22

定義されない。

答え23

① a/b,② a%b

答え24

「%d」のところが10進整数とされてしまう。%%dとするとよい。

答え25

評価値(評価した結果としての右辺値)が異なる。

++iは「+を先に」

と覚えれば、インクリメント後のi(つまり1つ大きくなったi)が評価値となることは容易に想像つく。同じように、

i++は、「+は後回し」

と覚えれば、インクリメント前のi(つまり1つ大きくなる前のi)が評価値となることもわかる。

答え26

+=である。-はbについているものなので違う。

a=a+(-b)に同じ。

箱aに、箱aの中の紙と、箱bの中の紙の正負を入れ替えたものを足したものを入れる。

このような例を考えれば、複合代入演算子で、=は常に右に来るのを覚えるのも苦ではないだろう。

答え27

「式;」のこと。

i = i + 1は式、

i = i + 1;は文であり、式文。

答え28

a=(b=c)、こういう風に右ファーストなものを「結合性が右」という。

(a+b)-c、こういう風に左ファーストなものを「結合性が左」という。

答え29

1位

++(後置増分演算子、使い方は「x++」)

2位

++(前置増分演算子、使い方は「++x」)

&(アドレス演算子、使い方は「&x」)

\*(間接演算子、使い方は「\*x」)

+(単項+演算子、使い方は「+x」)

!(論理否定演算子、使い方は「!x」)

3位

\*(2項\*演算子、使い方は「x\*y」)

%(使い方は「x%y」)

4位

+(2項+演算子、使い方は「x+y」)

5位

<

<=

6位

==

!=

7位

&(ビット単位のAND演算子、使い方は「x&y」)

8位

&&

9位

||

10位

? :

11位

=

+=

答え30

num=num+1の値は2となる。(格納される値が式の値)

しかし、inum=dnumの方は、3となる。(格納される値が型変換されて、式の値)

答え31

(double)x

## 第三回講義

問題32

int i=1;

a[i]=i++;

は、=演算子の結合性から考えて、i++が先にされてからa[i]を評価するので、

a[2]=1;

となりそうだ。

しかし、そうとはいない。どういうことか。

問題33(独自見解。参考にするには検証してください)

次のやり方では、7が表示されるのか8が表示されるのか、はたまた意味不明な何かになるのかわからない。

int i = 7;

printf("%d\n",i=i++);

そこで、次のようにすることで、確実に8を表示させることができる。理由を説明せよ。

int i = 7;

printf("%d\n",i=(i++,i=i));

問題34

C言語において、偽とは整数0だが、真とは何か。1、非0という言葉を用いて説明せよ。

問題35

if文を用い、==を使わず、整数xの偶奇を調べるプログラムの、if文をかけ。

問題36

num%3が1の時、実行される文をいえ。また、文1のみ実行されるようにするにはどうしたらいいか述べよ。

switch(num%3)

{

case 0: 文0

case 1: 文1

case 2: 文2

}

問題37

if(x=3) printf(“xは3”);としてしまうと、xが何であろうと「xは3」と表示されてしまう。このようなミスが発生したときに、エラーが起こることでそのミスを発見しやすくなる方法とその方法名を述べよ

問題38

次のコードに{ }をつけて読みやすくせよ。また「?」部も適切に書き換えよ

if(a>=b)

if(a==b)

printf(“a==b\n”);

else

printf(“a?b\n”);

答え32

評価順序と結合性、優先順位は必ずしも一致しないということ。

答え33

(i++,i=i)の「,演算子」によって、**副作用完了点を無理やり与えている**。

但し、この演算子は「x,y」という形をとるから、yに当たる部分に無意味無害な「i=i」を与え、コンパイラをやり過ごしている。これにより、右辺は確実にインクリメントされたものとして評価される。

答え34

評価された値としての真(つまり、プログラムが俺たちにくれる真)は整数1であるが、ユーザが与える真は、非0なら何でもよい。

答え35

if(x%2)

{

printf(“奇数”);

}

else

{

printf(“偶数”);

}

解説:式では「!=0」が省略されていると考えるとよい。「!=0」は「は偽ではない」と全く同じことである。

答え36

文1と文2が実行される。文1だけにするためには、文1直後にbreak;をかく。

答え37

3==xのように書くようにする。そうすると、3=xと書いてしまったときに、エラーが返るので、ミスを発見しやすい。ヨーダ記法。

答え38

if(a>=b)

{

if(a==b)

{

printf(“a==b\n”);

}

}

else

{

printf(“a<b\n”);

}

## 第四回講義

問題39

数学で1<=x<=9といえば、xが1以上かつ9以下ということであるが、C言語でこれを言うとどうなるか

問題40

break文とcontinue文を説明せよ。

問題41

for文内でcontinue文を用いたとき、for(;;ここ)の処理は行われるか否か。

問題42(これは考えずともパッと答えられるよう、覚えよ)

int i=1;

do

{

printf(“%d番目\n”,i)

}while(i<=n)

としたとき、「何番目」まで表示されるか。また、この分の直後に於いてiに格納されている値を言え。

答え39

1<=xが評価され、１または0になる。いずれにしても明らかに9以下だから、常に真(1)が返る。

答え40

break文は処理の流れを強制的に止め、そのブロックから抜ける。

continue文は、繰り返し文内の処理を飛ばし、ブロックの先頭位置に戻って次の処理を続ける。

答え41

行われる。

答え42

n番目まで。また、直後ではiにはn+1が格納されている。

## 去年の過去問

問題43

穴埋めせよ。

#include<stdio.h>

int main(){

int num1,num2,tmp;

do{

printf("1個目の正の整数:");

scanf("%d",&num1);

} while(num1 <= \_\_ア\_\_);

do{

printf("2個目の正の整数:");

scanf("%d",&num2);

} while(num2 <= \_\_イ\_\_);

while(num2){

tmp=num2;

\_\_ウ\_\_

num1=tmp;

}

printf("最大公約数は%d¥n",\_\_エ\_\_);

}

問題44

穴埋めせよ。

int main(){

int num,i;

int flag=1;

do {

printf("2以上の整数:");

scanf("%d",&num);

} while(num <= 1);

for(i=2;i<num;i++)

\_\_穴\_\_

if(flag) printf("素数¥n");

else printf("合成数¥n");

}

問題45

穴埋めせよ

#include<stdio.h>

int main(){

int num,i;

do {

printf("2以上の整数:");

scanf("%d",&num);

} while(num <= 1);

for(i=2;i<num;i++)

if(\_\_穴\_\_) break;

printf("最小素因数:%d¥n",i);

}

問題46

穴埋めせよ

#include<stdio.h>

int main(){

int i;

char str1[100];

char str2[100];

printf("文字列を入力:");

scanf("%s",str1);

for(i=0; \_\_ア\_\_; i++)

\_\_イ\_\_

str2[i]='¥0';

printf("コピーされた文字列:%s¥n",str2);

}

問題47

穴埋めせよ

#include<stdio.h>

int main(){

int i;

char str1[100];

printf("文字列を入力:");

scanf("%s",str1);

printf(“逆順にすると、”);

for(i=0;\_\_穴\_\_;i++) ;

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

printf("%c",str1[i]);

}

答え43

ア、イ・・・正の整数なわけだから。0以下ではなくなるまで繰り返さなければいけない。よって共に0。

ウ、エ・・・

ユークリッドの互除法は次のようなものである。

128と56の最大公約数は

128%72==56

72%56==16

56%16==8

16%8==0

8%0==？(ここは実行されない)

よって8

さて、プログラムを見てみよう。

num1%num2を繰り返しているのか、もしくはnum2%num1かもしれないが、while(num2)に注目すると、num1%num2を繰り返しているとわかる。

128%72==56

72%56==16を考える。特に72%56==16は「新しい」と呼ぶことにする。

新しいnum2=num1%num2と、

新しいnum1=古いnum2が繰り返されている。

ここで前者はnum2=num1%num2でよいのだが、

後者はnum1=num2としてはいけない。(直前でnum2に代入しなおしたわけだから、値が変わってる。)

だから、予め古いnum2をtmpに入れておいたわけだ。従ってnum1=tmpとすればよい。

よって、ウはnum2=num1%num2となり、エはnum1となる。

答え44

プログラムを見ていると、!flagだったら合成数とある。

つまりflag==0なら合成数である。

ここから、flagは「numを何らかの数字で割った余り」ではないかと類推したい。何らかの数字は、for文で１ずつ変化するiでよかろうから

flag%i;

とまずかける。

しかし、それだけだと弱い。折角これが0になって合成数だとわかっても、iがまた変わってしまえば素数として判定されてしまう。だから、break文で抜ける必要がある。

よって

{

flag=num%i;

if(!flag)

break;

}

としたい。ブロックを使わずに書きたい場合、次のようにする。

if(!(flag=num%i)) break;

答え45

!(num%i)

答え46

ア・・・str1[i]!=‘\0’

null文字\0(日本語でいう「。」に当たるもの)はint型では0なので、偽となる。そのため単にstr1[i]としてもよい。

イ・・・str2[i]=str1[i]