プログラミングⅠ

report#1

提出日：5 月 9 日(木)

所属：工学部情報工学科

学籍番号：135713B

氏名：天願 寛之

目次.

1. 出力するメッセージを変更せよ。

1-a. 全角文字と半角文字、❤記号の混合文の場合

1-a-1. 高水準コードの一部

```
01 #include <stdio.h>
02 int main(){
03   printf("my heart❤ is 夏模様¥n");
04   return(0);
05 }
```

1-a-2. 実行結果

```
my heart❤ is 夏模様
```

1-a-3. 考察

イ. 半角文字も全角文字も、そして❤記号も正常に出力することが出来た。

1-b. 色々な文字種で試してみた場合

1-b-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("/%b/()":;"*@+>?{}`¥n");
  return(0);
}
```

1-b-2. 実行結果

```
helloworld.c:7:18: error: expected ')'
  printf("/%b/()":;"*@+>?{}`¥n");
                 ^
helloworld.c:7:9: note: to match this '('
  printf("/%b/()":;"*@+>?{}`¥n");
        ^
helloworld.c:7:20: warning: unknown escape sequence '¥*'
  printf("/b/()":;"¥*@+>?{}`¥n");
                  ^~
helloworld.c:7:32: error: extraneous ')' before ';'
  printf("/%b/()":;"*@+>?{}`¥n");
                               ^
helloworld.c:7:20: warning: expression result unused [-Wunused-value]
  printf("/%b/()":;"*@+>?{}`¥n");
                  ^~~~~~~~~~~
1 warning and 2 errors generated.
```

1-b-3. エラーの意味

イ. 7行目１８桁、‘)’入ると予期される

ロ. 7行目２０桁、知られていないエスケープシークエンスがある。

ハ. 7行目３２桁、ダブルクォーテーションの前の‘)’は不必要である

ニ. 7行目２０桁、 ”*@+>?{}`¥n” の部分が結果的に使われていない

1-b-4. 考察

イ. 正常にコンパイルする為にエラーの原因となっている文字を削除する。

ロ. ‘￥’プラス一文字があるとエスケープシークエンスとみなされるが、有効なものでなければならない為、削除する。

ハ. ダブルクォーテーションで囲まれた部分が３つあるとコンパイル出来ない部分が出でくるため（“ ¥n”）で囲まれたダブルクォーテーションを削除する。

1-b-5. 修正後

```
#include <stdio.h>
int main(){
  printf("/%b/():;*@+>?{}`¥n");
  return(0);
}
```

1-b-6. 実行結果

```
helloworld.c:7:13: warning: invalid conversion specifier 'b'
      [-Wformat-invalid-specifier]
  printf("/%b/():;*@+>?{}`¥n");
          ~^
1 warning generated.
```

1-b-7. エラーの意味

イ. ７行１３桁、変換指定詞‘b’が無効になっている。

1-b-8. 考察

イ. 正常にコンパイルできない文字を削除する。

1-b-9. ‘b’の削除

修正後 b と同様にエラーの原因となった”スラッシュ”、”(“、”)”、” : ”.....削除していくと‘￥n’も同様に変換指定が無効になるという現象が起きたため、修正過程でこれらの文字の直前にあった%が変換指定を無効にしていると考えられる。

従って、一回目の修正後のテキストから％だけを削除してみる。

1-b-10. 修正後

```
#include <stdio.h>
int main(){
  printf("/b/():;*@+>?{}`¥n");
  return(0);
}
```

1-b-11. 実行結果

```
/b/():;*@+>?{}`
```

1-b-12. 考察

イ. 正常にコンパイルすることが出来た。

ロ. ‘%’は直後の文字の変換指定を無効にすることが分かる。

ハ. (“¥n”)内であればほとんどの文字が正常にコンパイルすることが出来ると分かる。

ニ. ‘¥’とダブルクォーテーションはプログラムの表記に必要なものであるためコードの表記が間違っていると認識されている。

２. 同じメッセージを３回、別々の行に出力せよ

2-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C World!.¥nHello, C World!.¥nHello, C World!.¥n");
  return(0);
}
```

2-a-2. 実行結果

```
Hello, C World!.
Hello, C World!.
Hello, C World!.
```

2-a-3. 考察

イ. 同じメッセージを３回、別々の行に出力することが出来た。

ロ. 特殊文字‘¥n’により改行が行われていると分かった。

ハ. 特殊文字の前後に別の文字が繋がっていても特殊文字として認識されている。

2-a-4. printf 関数を３つ用い、２つ目以降の‘¥n’を抜いてみる

```
#include <stdio.h>
int main(){
  printf("Hello, C World!.¥n");
  printf("Hello, C World!.");
  printf("Hello, C World!.");
  return(0);
}
```

2-a-5. 実行結果

```
Hello, C World!.
Hello, C World!.Hello, C World!.%
```

2-a-6. 考察

イ. 複数の printf 関数を用いても別々の行に出力することが可能と分かる。

ロ. 複数の printf 関数を用いた場合、‘%’が出てきてしまった。

ハ. 直前の printf 関数で‘¥n’が出力されたことで、次の printf 関数の出力は改行して行われた。

3. 「Hello,」と「C　World!」を別々の行に出力せよ。

高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello,¥n C World!.¥n");
  return(0);
}
```

実行結果

```
Hello,
 C World!.
```

考察

イ. 正常に改行する事が出来た。

4. printf("…")と printf("…¥n")の違いについて延べよ。

例題 hello.c より￥n を消去し結果を比較する。

4-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C World!");
  return(0);
}
```

4-a-2. 実行結果

```
Hello, C World!%
```

4-a-3. 考察

イ. 2−b、2−c により‘￥n’が挿入された部分で改行が起こる事は分かっているが、一文の出力では改行の意味はない。

ロ. 例題 hello.c の実行結果とは異なり、後尾に‘%’が付いてしまった。

ハ. もし、‘¥n’が文の最後以外にあるとどうなるか試みてみる。

4-a-4. 修正後

```
#include <stdio.h>
int main(){
  printf("Hello, ¥nC World!");
  return(0);
}
```

4-a-5. 実行結果

```
Hello,
C World!%
```

4-a-6. 考察

イ.‘￥n’により改行したが、結局後尾に‘%’が付いてしまった。

ロ. 複数の printf 関数を使った出力のとき、２つ目以降の printf 関数に‘￥n’を挿入しなかった結果、出力結果の後尾に‘%’が付いてしまっているが、２つ目と３つ目の間には無いことが分かる。

ハ. ３つ目の printf 関数だけに‘￥n’があるとどうなるか調べる。

4-a-7. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C World!");
  printf("Hello, C World!");
  printf("Hello, C World!¥n");
  return(0);
}
```

4-a-8. 実行結果

```
Hello, C World!Hello, C World!Hello, C World!
```

4-a-9. 考察

イ. 最後の文末に‘￥n’があれば‘%’が文の後尾に付かないとわかる。

ロ. printf("…")と printf("…¥n")の違いについて、‘￥n’による改行の効果は無いが、‘%’が現れるかどうかの違いがでる。

5. 同じメッセージを 3 回、同一行に出力せよ。

5-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C World! Hello, C World! Hello, C World!¥n");
  return(0);
}
```

5-a-2. 実行結果

```
Hello, C World! Hello, C World! Hello, C World!
```

5-a-3. 考察

イ. ‘%’が現れないように文末のダブルクォーテーションの前に‘￥n’を挿入することにより同じメッセージを３回、同一行に出力することが出来た。

ロ. また、３つの printf 関数を使っても出力可能である。

6. 次のような菱形模様(「*」を用いる)を出力せよ。

```
  *
 ***
*****
 ***
  *
```

6-a. １つの printf 関数で出力してみる。

6-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("  *¥n ***¥n*****¥n ***¥n  *¥n");
  return(0);
}
```

6-a-2. 実行結果

```
  *
 ***
*****
 ***
  *
```

6-a-3. 考察

イ．１つの printf 関数の時、‘*’の直前に適度なスペースを空けたり‘￥n’を用いて改行することで、出力したい模様を出力することが出来た。

6-b. 複数の printf 関数で出力してみる。

6-b-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("  *¥n ");
  printf(" ***¥n");
  printf("*****¥n");
  printf(" ***¥n");
  printf("  *¥n");
  return(0);
}
```

6-b-2. 実行結果

```
  *
 ***
*****
 ***
  *
```

6-b-3. 考察

イ．複数の printf 関数を用いて出力することが出来た。

ロ．同様に適度なスペースを空けたり、‘￥n’を用いて改行することで、出力したい模様を出力することが出来た。

7. 「*」を用いて、自分の好きな形を出力せよ。

7-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("                  ****¥n");
  printf("               *************      ***¥n");
  printf("            ******************  *******¥n");
  printf("     ***   ********************* ********¥n");
  printf("    ******* ************************ *****¥n");
  printf("   ******* ******************  ****** ****¥n");
  printf("   ****** *******  ******************* ***¥n");
  printf("    **** ****************************** ***¥n");
  printf("       * ****************************** ****¥n");
  printf("     *** ***************************** ******¥n");
  printf("    ***** **********        *******¥n");
  printf("    ****** ************************¥n");
  printf("    *       *********************¥n");
  printf("               ***************¥n");
  printf("                     ********¥n");
  printf("                     *********¥n");
  printf("                     *********¥n");
  printf("                     **********¥n");
  printf("                     ***********¥n");
  printf("                     ***      **¥n");
  printf("                     *          *¥n");
  return(0);
}
```

7-a-2. 実行結果

```
                  ****
              *************      ***
            ******************  *******
     ***   ********************* ********
   ******* ************************ *****
  ******* ******************  ****** ****
  ****** *******  ******************* ***
   **** ****************************** ***
      * ****************************** ****
    *** ***************************** ******
    ***** **********        *******
    ****** ************************
    *       *********************
               ***************
                     ********
                     *********
                     *********
                     **********
                     ***********
                     ***      **
                       *          *
```

7-a-3. 考察

イ. 複数の printf 関数を用いる方法で入力した。

ロ. 意図した図を出力することが出来た。

8. テキスト PP.50【特殊な文字(エスケープシーケンス)】について考察せよ。

8-a. バックスペース‘￥b’について

8-a-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥b World!¥n");
  return(0);
}
```

8-a-2. 実行結果

```
Hello,  World!
```

8-a-3. 考察

イ. ‘￥b’の直前にあった‘C’が消えてしまった。

ロ. バックスペースの機能は直前の文字を消す事である。

ハ. ‘H’の直前と‘n’の直後、また‘￥n’の直前に挿入してみる。

8-a-4. 修正後

```
#include <stdio.h>
int main(){
  printf("¥bHello, C World! ¥b¥n¥b");
  return(0);
}
```

8-a-5. 実行結果

```
Hello, C World!
```

8-a-6. 考察

イ. 例題 hello.c と同じ結果が得られた事､かつ､8-a-1 で得られた結果と比較することにより、‘H’の直前の‘￥b’は効果がなく、‘n’の直後の‘￥b’も‘n’を消す事はなく、‘￥n’として出力されたと分かる。何故なら、‘%’が現れなかった為である。さらに、‘！’も出力されている。

ロ. ‘￥b’は直前の文字を消すが、特殊文字‘￥n’の直前直後では効果を発揮しないと分かった。

8-b. フォームフィード‘￥f’について

8-b-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥f World!¥n");
  return(0);
}
```

8-b-2. 実行結果

```
Hello, C
        World!
```

8-b-3. 考察

イ．‘￥f’後の文が次行同文字目に移動した。

ロ．上の結果に h−１と同様の位置に‘￥f’を挿入してみる。

8-b-4. 修正後

```
#include <stdio.h>
int main(){
  printf("¥fHello, C¥f World!¥f¥n¥f");
  return(0);
}
```

8-b-5. 実行結果

```
Hello, C
        World!
```

8-b-6. 考察

イ．先の結果に３つの空白行が加わっている。

ロ．‘￥n’１つではこのような空白行の出力は起きないため３つの‘￥f’が出力されたと分かる。

ロ．‘￥b’とは異なり、特殊文字の直前直後でも出力されると分かった。

ハ．‘￥f’について、後の文を次行同文字目に移し、特殊文字や位置に関係なく出力されることが分かった。

8-c. 改行‘￥n’について

8-c-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥n World!¥n");
  return(0);
}
```

8-c-2. 実行結果

```
Hello, C
 World!
```

8-c-3. 考察

イ．適当な位置に‘￥n’を挿入した結果、その直後でされている。

ロ．文の最初に挿入して、さらに２つ連続で並べみる。

8-c-4. 修正後

```
#include <stdio.h>
int main(){
  printf("¥nHello, C World!¥n¥n");
  return(0);
}
```

8-c-5. 実行結果

```

Hello, C World!

```

8-c-6. 考察

イ．文の最初の‘￥n’と２つ連続して並んだ‘￥n’の先方のほうにより改行がおこなわれたことが分かった。

ロ．文末の‘￥n’以外、場所に関係なく出力されることが分かった。

8-d. リターン‘￥r’について

8-d-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥r World!¥n");
  return(0);
}
```

8-d-2. 実行結果

```
  World!C
```

8-d-3. 考察

イ．‘C’から以前の文がなくなり、‘C’のみが‘！’の直後に移動した。

ロ．よくわからない為、他の場所に挿入してみる。

8-d-4. 修正後

```
#include <stdio.h>
int main(){
  printf("Hello, C W¥rorld!¥n");
  return(0);
}
```

8-d-5. 実行結果

```
orld!, C W
```

8-d-6. 考察

イ. 再試行結果に‘,’があることから、‘W’以降の文字が文の始めから同じ文字数分押しのけて、移動したと分かる。

ロ. ‘￥r’以前の文字数より以降の文字数が多い時どうなるか調べる。

8-d-7. 修正後

```
#include <stdio.h>
int main(){
  printf("Hell¥ro, C World!¥n");
  return(0);
}
```

8-d-8. 実行結果

```
o, C World!
```

8-d-9. 考察

イ. ‘￥r’以前の文字だけが消えてしまった。以降の文が押しのけている。

ロ. ‘￥n’の直前直後にあるとどうなるか調べる。その際‘￥r’の効果が見られない場合、直後に文字が無い為か、‘￥n’があるためか判断する為に文中に‘￥n’を挿入し、その直前直後に‘￥r’を挿入する。

8-d-10. 修正後（直後）

```
#include <stdio.h>
int main(){
  printf("Hello, C¥n¥r World!¥n");
  return(0);
}
```

8-d-11. 実行結果

```
Hello, C
 World!
```

8-d-12. 考察

イ. ‘￥n’により改行しただけと分かった。

8-d-13. 修正後（直前）

```
#include <stdio.h>
int main(){
  printf("Hello, C¥r¥n World!¥n");
  return(0);
}
```

8-d-14. 実行結果

```
Hello, C
 World!
```

8-d-15. 考察

イ．‘￥n’により改行しただけと分かる。

ロ．‘￥n’の直前直後でも結果が変わらないことにより、‘￥r’は効果を発揮出来なかったと分かる。

ハ．‘￥r’について、‘￥r’以降の文を文の始めに移動すると分かった。その際、移動した文字数分押しのけることも分かった。

ニ．‘￥n’の直前直後では、効果が発揮出来ないことも分かった。

8-e. タブ‘￥t’について

8-e-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥t World!¥n");
  return(0);
}
```

8-e-2. 実行結果

```
Hello, C             World!
```

8-e-3. 考察

イ．‘￥t’を挿入した部分に８文字分の空白が追加された。

ロ．別の部分に挿入した結果、１～７文字分の空白も出来た。

ハ．空白は１～８文字目に挿入するにつれ、８～１文字分と少なくなり、また９文字目から８文字分の空白が出来るようになった。

ニ．‘￥n’の直前直後にあるとどうなるか調べる。h−４と同じ理由で文中に挿入した‘￥n’で試す。

8-e-4. 修正後（直前）

```
#include <stdio.h>
int main(){
  printf("Hello, C¥t¥n¥t World!¥n");
  return(0);
}
```

8-e-5. 実行結果

```
Hello, C
            World!
```

8-e-6. 考察

イ．‘￥n’により改行してしまい直前の‘￥t’の効果が見えないが、‘￥n’の直後の‘￥t’により８文字分空いていることが分かった。

ロ．これまでの考察より‘￥n’の直前直後ではどちらでも発揮出来るか、どちらでも出来ないかの２通りしかないことから、‘￥n’の直前直後ではどちらでも‘￥t’は正常に出力されていると推測することが出来る。

ハ．‘￥t’について１～８文字分の空白を加えることが出来る。また、’￥n‘の直前直後でも効果が発揮できると思われる。そして、何文字分の空白が出来るかは、何文字目に挿入するかによることが分かった。

8-f. アポストロフィー‘¥’’について

8-f-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥' World!¥n");
  return(0);
}
```

8-f-2. 実行結果

```
Hello, C' World!
```

8-f-3. 考察

イ．挿入した位置にアポストロフィーが出力することが出来た。

ロ．挿入する位置を文頭、‘￥n’の前後で試みる。

8-f-4. 修正後

```
#include <stdio.h>
int main(){
  printf("¥'Hello, C World!¥'¥n¥'");
  return(0);
}
```

8-f-5. 実行結果

```
'Hello, C World!'
'%
```

8-f-6. 考察

イ．‘%’が現れるのは前述の考察で予想される結果であったため今回は触れない事とする。以後も同様である。

ロ．アポストロフィーを挿入した位置にアポストロフィーが出力された。

ハ．‘￥n’が有る無しに関わらず出力される。

ニ．バックスラッシュなしで出力してみる。

8-f-7．修正後

```
#include <stdio.h>
int main(){
  printf("'Hello, C World!'¥n'");
  return(0);
}
```

8-f-8．実行結果

```
'Hello, C World!'
'%
```

8-f-9．考察

イ．バックスラッシュなしでも同様の結果が得られた。また、この場合も位置に関係は無いと分かる。

8-g．ダブルクォーツ（￥“）について

8-g-1．高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥" World!¥n");
  return(0);
}
```

8-g-2．実行結果

```
Hello, C" World!
```

8-g-3．考察

イ．挿入した部分にダブルクォーツが出力された。

ロ．挿入する位置を文頭、‘￥n’の前後で試みる。

8-g-4．修正後

```
#include <stdio.h>
int main(){
  printf("¥"Hello, C World!¥"¥n¥"");
  return(0);
}
```

8-g-5．実行結果

```
"Hello, C World!"
"%
```

8-g-6. 考察

イ．ダブルクォーツを挿入した位置にダブルクォーツが出力された。

ロ．‘¥n’が有る無しに関わらず出力された。

ハ．バックスラッシュなしで出力してみる。

8-g-7. 修正後

```
#include <stdio.h>
int main(){
  printf(""Hello, C World!"¥n"");
  return(0);
}
```

8-g-8. 実行結果

```
helloworld.c:7:12: error: expected ')'
  printf(""Hello, C World!"¥n"");
           ^
helloworld.c:7:9: note: to match this '('
  printf(""Hello, C World!"¥n"");
        ^
helloworld.c:7:31: warning: missing terminating '"' character
      [-Winvalid-pp-token]
  printf(""Hello, C World!"¥n"");
                               ^
1 warning and 1 error generated.
```

8-g-9. 考察

イ．予想通り、ダブルクォーツはバックスラッシュなしではエラーとなりうる原因となる。

ロ．バックスラッシュなしではプログラムにおいて意味をもつものとなり、正しい表記でなければ正常に出力することが出来ないと分かった。

8-h. バックスラッシュ（¥）について

8-h-1. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥¥ World!¥n");
  return(0);
}
```

8-h-2. 実行結果

```
Hello, C¥ World!
```

8-h-3. 考察

イ．バックスラッシュを挿入した位置にバックスラッシュが出力された。

ロ．挿入する位置を文頭、‘￥n’の前後で試みる。

8-h-4．修正後

```
#include <stdio.h>
int main(){
  printf("¥¥Hello, C World!¥¥¥n¥¥");
  return(0);
}
```

8-h-5．実行結果

```
¥Hello, C World!¥
¥%
```

8-h-6．考察

イ．バックスラッシュを挿入した位置でバックスラッシュが出力された。

ロ．‘￥n’の有る無しに関わらず出力された。

ハ．バックスラッシュなし（‘￥’単体）で出力してみる。

8-h-7．修正後

```
#include <stdio.h>
int main(){
  printf("¥Hello, C World!¥¥n¥");
  return(0);
}
```

8-h-8．実行結果

```
helloworld.c:7:10: warning: missing terminating '"' character
     [-Winvalid-pp-token]
  printf("¥Hello, C World!¥¥n¥");
         ^
helloworld.c:7:10: error: expected expression
1 warning and 1 error generated.
```

8-h-9．考察

イ．予想通り、バックスラッシュ（単体）はバックスラッシュなしではエラーとなりうる原因となる。

ロ．バックスラッシュなしではプログラムにおいて意味をもつものとなり、正しい表記でなければ正常に出力することが出来ないと分かった。

8-i．‘￥nnn’（nnn の部分には８進数の数字を入力する）について

（￥000~￥015）

8-i-1. 高水準コード

```
#include <stdio.h>
int main(){
  printf("Hello, C¥000 World!¥n");
  printf("Hello, C¥001 World!¥n");
  printf("Hello, C¥002 World!¥n");
  printf("Hello, C¥003 World!¥n");
  printf("Hello, C¥004 World!¥n");
  printf("Hello, C¥005 World!¥n");
  printf("Hello, C¥006 World!¥n");
  printf("Hello, C¥007 World!¥n");
  printf("Hello, C¥010 World!¥n");
  printf("Hello, C¥011 WOrld!¥n");
  printf("Hello, C¥012 WOrld!¥n");
  printf("Hello, C¥013 WOrld!¥n");
  printf("Hello, C¥014 WOrld!¥n");
  printf("Hello, C¥015 WOrld!¥n");
  return(0);
}
```

8-i-2. 実行結果

```
helloworld.c:7:19: warning: format string contains '¥0' within the string body
      [-Wformat]
  printf("Hello, C¥000 World!¥n");
         ~~~~~~~~~^~~~~~~~~~~~~~
1 warning generated.
```

8-i-3. 考察

イ．‘￥000’でエラーが起きたため、このコードを削除する

8-i-4. 修正後

```
#include <stdio.h>
int main(){
  printf("Hello, C¥001 World!¥n");
  printf("Hello, C¥002 World!¥n");
  printf("Hello, C¥003 World!¥n");
  printf("Hello, C¥004 World!¥n");
  printf("Hello, C¥005 World!¥n");
  printf("Hello, C¥006 World!¥n");
  printf("Hello, C¥007 World!¥n");
  printf("Hello, C¥010 World!¥n");
  printf("Hello, C¥011 World!¥n");
  printf("Hello, C¥012 World!¥n");
  printf("Hello, C¥013 World!¥n");
  printf("Hello, C¥014 World!¥n");
  printf("Hello, C¥015 World!¥n");
  return(0);
}
```

8-i-5. 実行結果

```
Hello, C World!
```

```
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello,  World!
Hello, C            World!
Hello, C
 World!
Hello, C
        World!
Hello, C
        World!
 World!C
```

8-i-6. 考察

イ. ‘￥000’を削除すると正常にコンパイルすることが出来た。

ロ. ‘￥001’~‘￥007’までは特になにも起きてない。

ハ. ‘￥010’では‘￥b’と同じ、バックスペースの効果があると分かる。

ニ. ‘￥011’では‘￥t’と同じ、タブの効果があると分かる。

ホ. ‘￥012’では‘￥n’と同じ、改行の効果があると分かる。

ヘ. ‘￥013’と‘￥014’では‘￥f’と同じ、フォームフィードの効果があると分かる。

ト. ‘￥015’では‘￥r’と同じ、リターンの効果があると分かる。

（‘￥016~￥032’）

8-i-7. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥016 World!¥n");
  printf("Hello, C¥017 World!¥n");
  printf("Hello, C¥020 World!¥n");
  printf("Hello, C¥021 World!¥n");
  printf("Hello, C¥022 World!¥n");
  printf("Hello, C¥023 World!¥n");
  printf("Hello, C¥024 World!¥n");
  printf("Hello, C¥025 World!¥n");
  printf("Hello, C¥026 World!¥n");
  printf("Hello, C¥027 World!¥n");
  printf("Hello, C¥030 World!¥n");
  printf("Hello, C¥031 World!¥n");
  printf("Hello, C¥032 World!¥n");
  return(0);
}
```

8-i-8. 実行結果

```
Hello, C World!
Hello, C World!
Hello, C World!
```

```
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
```

8-i-9. 考察

イ．特に効果が見られなかった。

（￥033~￥047）

8-i-10. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("Hello, C¥033 World!¥n");
  printf("Hello, C¥034 World!¥n");
  printf("Hello, C¥035 World!¥n");
  printf("Hello, C¥036 World!¥n");
  printf("Hello, C¥037 World!¥n");
  printf("Hello, C¥040 World!¥n");
  printf("Hello, C¥041 World!¥n");
  printf("Hello, C¥042 World!¥n");
  printf("Hello, C¥043 World!¥n");
  printf("Hello, C¥044 World!¥n");
  printf("Hello, C¥045 World!¥n");
  printf("Hello, C¥046 World!¥n");
  printf("Hello, C¥047 World!¥n");
  return(0);
}
```

8-i-11. 実行結果

```
helloworld.c:17:24: warning: invalid conversion specifier 'W'
     [-Wformat-invalid-specifier]
  printf("Hello, C¥045 World!¥n");
                  ~~~~~^
1 warning generated.
```

8-i-12. 考察

イ．教科書 p407 を参照した結果‘￥045’は‘%’と同じ効果を持つため、前述の考察通りエラーの原因となっていると分かる。

ロ．‘￥045’のコードを削除して再試行する。

8-i-13. 実行結果

```
Hello, CWorld!
Hello, C World!
Hello, C World!
Hello, C World!
Hello, C World!
```

```
Hello, C  World!
Hello, C! World!
Hello, C" World!
Hello, C# World!
Hello, C$ World!
Hello, C& World!
Hello, C' World!
```

8-i-14. 考察

イ. ‘￥033’では直後の文字を出力しない効果があると分かる。

ロ. ‘￥034’~‘￥037’までは特に効果は見られない。

ハ. ‘￥040’は1回分のスペースが出来ている事が分かる。

ニ. ‘￥041’~‘￥047’までは特定の一文字が出力されている。

8-i-15. （‘￥050’~）について

イ. ‘￥050’~‘￥176’までは特定の一文字が出力される結果となり、‘￥177’においては特に効果は見られなかった。

9. エラーについて考察せよ

9-a-1. 考察

イ. ダブルクォーテーションとバックスラッシュ、そして‘%’は単体では一記号としての出力が正常に行われないことが分かった。

ロ. ダブルクォーテーションとバックスラッシュはプログラムの表記に必要な文字であるため正確な表記でなければ出力できないことが分かる。

ハ. しかし、エスケープシーケンスとして出力すれば表記されることが分かった。

ニ. ‘%’に関しては直後にある文字の変換指定を無効にする効果があると分かり、これがエラーの原因となっていると分かった。よって、‘%’の表記を可能にする方法を試行する。

9-a-2. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("%%¥n");
  return(0);
}
```

9-a-3. 実行結果

```
%
```

エスケープシーケンスとして出力してみる。

9-a-4. 高水準コードの一部

```
#include <stdio.h>
int main(){
  printf("¥%¥n");
  return(0);
}
```

9-a-5. 実行結果

```
helloworld.c:7:13: warning: invalid conversion specifier '
' [-Wformat-invalid-specifier]
  printf("¥%¥n");
          ~~^
1 warning generated.
```

9-a-6. 考察

イ. エスケープシーケンスとしては出力出来なかったが、‘％％’と２回連続で入力することで出力することが出来ると分かった。

10. 感想

ロ. 手書きはキツイっす。