プログラミング演習 1 レポート A
—C 言語による構文解析の基本—

氏名: 原 直 (HARA, Sunao)
学生番号: 0941xxxx

出題日: 20xx 年 xx 月 xx 日
提出日: 20xx 年 xx 月 xx 日
締切日: 20xx 年 xx 月 xx 日

※執筆上の注意：本書は空想上の課題に対するレポートの執筆例である．章の構成と書くべき内容の参考として提示するものであるため，課題内容やプログラムの仕様などは，実際の演習課題の指示に従って適切にまとめ直す必要がある．文や文の一部を「・・・」や「??」によって省略している箇所があるが，これに穴埋めをするだけで，レポートが完成するわけではない．なお，サンプルと同じ書き出しで文章を書く必要もない．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

1 概要

※執筆上の注意：演習課題説明書を読み，本レポートとして実施した課題が分かるように書く．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

本演習では・・・を作成する・・・．・・・のため，・・・・をおこなう．

本レポートでは，演習中に取り組んだ課題として，以下の課題??から課題??についての内容を報告する．

課題 1　Linux 環境における C 言語プログラミング
課題 2　文字列操作・・・（**サンプルのため，省略**）

2 課題 1：Linux 環境における C 言語プログラミング

※執筆上の注意：少なくとも，(1) gcc によるコンパイル方法，(2) コンパイルされたプログラムを実行する方法については記載すること．
なお，本講義では，Red Hat Linux は使っていない．適切に書き直すこと．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

本講義では，Linux 環境における C 言語プログラミングをおこなう．そこで，本章では・・・についてまとめる．

以降のプログラムは，いずれも Red Hat Linux 3.2.2-5 で動作を確認しているが，一般的な UNIX で動作することを意図して作成している．なお，以降の実行例における，行頭の $ 記号は，Red Hat Linux 3.2.2-5 におけるターミナルのプロンプトである．

C 言語で書かれたソースコードは，gcc でコンパイルすることで，プログラムの実行ファイルを生成する．例えば，program1.c であれば，

```
$ gcc -Wall ・・・
```

のようにコンパイルする．この時，生成されるプログラムの実行ファイルは・・・という名前になる．

ここで，-Wall とは... するためのオプションであり，-o とは・・・を指定するオプションである．これらのオプションをつけることで，・・・（略）

生成されたプログラムを実行するには，以下のようにターミナルで入力する．

```
$ program1
```

3 課題 2：文字列操作の基礎

3.1 プログラムの説明

※執筆上の注意：変数や数値は \verb や$$ で囲って，適切な書体で記述することを忘れずに．また，章・節番号や図表番号の引用も，数字を直接書くのではなく，LaTeX の機能を使うこと．なお，このサンプルでは “わざと” 一部の処理を省略している．見た目の違いを確認して，自分のレポートでは処理を忘れないようにしよう．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

本課題で作成するプログラムは，・・・をするプログラムである．プログラムリストは・・・節に添付している．

プログラムは全部で 267 行からなる．まず，文字列操作関数として，subst() 関数を 11–25 行目で宣言し， split() 関数を 27-44 行目で宣言している． 次に，これらの関数の動作を確認するため，・・・（省略）最後に，・・・において，・・・（省略）

subst(STR, C1, C2) 関数は，STR が指す文字列中の，文字 C1 を文字 C2 に置き換える関数である．プログラム中では，xxx 関数の中で，入力文字列中の末尾に付く改行文字をヌル文字で置き換えるために使用している．

split(STR, ...) 関数は，・・・（※サンプルのため，いろいろと省略）

3.2 動作確認の方法

※執筆上の注意：プログラムの使用法を説明する節である．入力データの準備，コマンドの実行方法，実行中のキーボード入力例，などを書く．ソースコードのファイルと，このレポートの記述通り実行すれば，報告者と同じ結果が再現できるように書いておく，という意識が必要である．なお，入力データの準備が不要なら無理に何かを準備する必要はない．このサンプルファイルでは，テンプレート文ではなく，架空の課題に対する記載例を示しているに過ぎない．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

本課題で作成したプログラムは，・・・をするプログラムであり，・・・が必要である．そこで，まずは次の方法で・・・を準備する．

（※サンプルのため，いろいろと省略）

次に，以下の方法でプログラムを実行する．

（※サンプルのため，いろいろと省略）

実行結果として，次のような出力が得られる．

（※サンプルのため，いろいろと省略）

3.3 動作確認の結果

※執筆上の注意：動作確認の観点から結果を述べる．すなわち，前節で記述した実行方法（や入力データ）と実験結果から，課題プログラムの動作として妥当かどうかの言及をする．想定外の動作を発見したなら，そのことも書いておくとよい．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

前節における，入力 1 に対する結果 1 を見ると，・・・（※サンプルのため，いろいろと省略）

入力 3 のように・・・では，結果 3 のような・・・となることがある．そのため，・・・関数を利用する際にｘｘｘとならないように，注意する必要がある．

（※サンプルのため，いろいろと省略）

4 課題 3：標準入力を受け付ける構文解析プログラムの作成

4.1 プログラムの説明

（※サンプルのため，いろいろと省略）

4.2 動作確認の方法

（※サンプルのため，いろいろと省略）

4.3 動作確認の結果

（※サンプルのため，いろいろと省略）

5 感想

（※サンプルのため，いろいろと省略）

6 作成したプログラムのソースコード

※執筆上の注意：余白にはみ出さないように注意．ソースコードは，1 行あたり 80 文字以内に収めるのが無難．また，??章のような状態でレポートを提出しないように注意すること．（この注意書きは提出用のレポートでは必ず消すか，コメントアウトすること）※

6.1 課題 2 のソースコード

課題 2 のために作成したプログラムを以下に添付する．なお，1 章に示した課題については，??章で示したようにすべて正常に動作したことを付記しておく．

```
1: #include <stdio.h>
2:
3: int main()
4: {
5:     char s[4] = {'A', 'B', 'C', '\0'};
```

```
 6:     printf("s = %s\n", s);
 7:     return 0;
 8: }
 9: // 以下は，はみ出し文字数を確認するためのダミーコメントです．
10: //00000000111111111122222222223333333333444444444455555555556666666666777777777788888888889999999
```

6.2 課題 3 のソースコード

課題 3 のために作成したプログラムを以下に添付する．なお，1 章に示した課題については，**??**章で示したようにすべて正常に動作したことを付記しておく．

```
1: #include <stdio.h>
2:
3: int main()
4: {
5:     char s[4] = {'A', 'B', 'C', '\0'};
6:     printf("s = %s\n", s);
7:     return 0;
8: }
```