参考資料１

プログラム作成実習における提出用紙の工夫

（作成時）宮城障害者職業能力開発校

（現　在）塩釜高等技術専門校

砂沢　恵子

はじめに・・・

今回の研修で取り上げられた品質管理の手法そのままを直接担当している学科での訓練に持ち込む前段階として、現在のプログラム実習のやり方を改善し、その考え方を訓練生の身につけさせる工夫を考えてみたい。

担当学科のプログラム実習訓練の現時点での問題点を挙げてみると

１．生徒数が少ないため、比較的担任の目は届きやすい。そのため、実習中でもエラー箇所があれば、生徒自身が気付く前に指摘してしまうことがある。

２．障害を持っているためか、作業の進捗が他の生徒より遅れても余り気にしない傾向がみられる。差別感や疎外感を抱かせずに作業要領の悪さを指摘する必要を常々感じている。

３．実習に与える課題はステップ数にすればせいぜい数十の長さのものであり、アルゴリズムは授業で練習したものを２、３組み合わせる程度であるが、それでも最初から大きな失敗なく流れ図が作れるのは２分の１程度であり、これらの生徒が課題を仕上げる頃になっても、課題のテーマが理解できない生徒もいる。

4．現在、実習時に提出させているのは以下のとおりである。

（1）提出用紙（B４版　下図のような形式）

（2）フローチャート

（3）スペーシングチャート

（4）プログラムリスト

（5）実行結果

課題名＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿　提出日＿＿＿＿＿＿

作成者＿＿＿＿＿＿

（処理の概要）

（出力形式）；スペーシングチャートのとおり

（入力形式）

（流れ図）；フローチャートのとおり

改善点

1．現状の提出用紙に付け加え、手順確認用紙の記入を通し、プログラム作成作業の各段階に対しての確実なアプローチを自己管理することを習慣づけたい。

(→作業手順確認用紙)

2．エラーの記録を確実にとらせ、自己の弱点把握とエラーの再発防止に役立てる。

(→文法エラーリスト)

3．完成したプログラムに対して、品質管理シートを記入させソフトウェアの品質に対する感覚を養いたい。

(→プログラム品質管理シート)

4．訓練途中で生じる失敗結果についても、そのときに実行させたプログラムリストとともに保存させる。

作業手順確認用紙

課題が出されたのは		月　日	校時
スペーシングチャートの完成日 (　〃　に要した時間数　時間　)			
コーディング	開始日	月　日	校時
〃	終了日	月　日	校時
プログラム入力	開始日	月　日	校時
〃	終了日	月　日	校時
データ入力	開始日	月　日	校時
〃	終了日	月　日	校時
デバッグ	開始日	月　日	校時
〃	終了日	月　日	校時

今回の課題で参照又は使用したプログラム；

教科書　　P.　　～　　P.

以前作成のプログラム名

〃　　データファイル名

プログラム管理シート

チェック項目		チェック日
プログラムの見やすさ		
一目でわかるか	作成日付	
	作成者	
	課題名・テーマ	
注釈行の活用	主プログラムの処理内容	
	副プログラム　〃	
字下げ	条件判断の箇所	
	繰り返しの箇所	
名前のわかりやすさ		
データ名	入力データ名の特徴づけ	
	出力データ名　〃	
	他の変数名　〃	
処理・段落名		
出力結果の見やすさ		
余白	上・下・右・左	
題名	目立ち度	
項目名	間隔	
	データとの並び具合	
明細データ	行間隔	
	正・負符号・円記号・カンマ	

おわりに。。。

短時間で思いつくまま作成したので、書式等帰校後十分に練り直し、訓練に活用したい。

学生に実習で提出させたプログラムの品質評価項目として、昨年度の研修で作成した中にも、また授業の中でも、プリントアウトの結果については余りなかったようだが、実際の訓練のレベルでは必要と思い入れてみた。

今回の研修では講師陣の充実ぶりに感心し、どの講義も興味深く聞かせていただいた。まさに研修の醍醐味という感じで、コーディネートしてくださった保田先生に感謝したい。

（平成５年１２月作成）

参考資料２

情報処理実習における
ソフトウェア管理手法について

神奈川県立藤沢高等職業技術校
久　保　雅　俊

1 ソフトウェア管理に関する教材の目的

各コンピュータ企業では、独自の方法によって自社のソフトウェアの品質管理を行っている。ところが、その会社内での事情や開発体制によって、進行の仕方や帳票内容が異なり、職業訓練ではその内容をそのまま活かすことはできない。そこで、どのように整理したら即戦力となるような教材ができあがるか考えてみた。

2 指導体制

教材内容は、生徒からの自由課題とする。ただし、訓練期間が短い場合（例えば１か月）などの時には作成時間がなくなるので、こちらから課題内容を提示する。これについての詳細は後述する。

指導体制は、自由課題ということもあり、生徒数分の課題ができあがるので、指導員一人では到底対処できない。そこで、生徒ごとに担当指導員を決め、進捗状況の管理、課題の相談などは、その指導員が面倒をみる。これでは、生徒内に派閥ができるのではという心配もあろうが、小さな問題についての質問は、どの指導員でも受け付けるようにするなど、十分配慮すれば問題はないであろう。

特に、このようにしなければならないもう一つの理由として、指導員間の意見の相違というものが挙げられる。ある指導員がよいと思ってアドバイスしたことも、他の指導員から違うように指導され、生徒自身が混乱するような場面を何度も見てきた。そこで、誰が答えても同じとなるような質問には誰でも答え、そうでないものについて、例えば、課題そのものの進め方、使用する帳票の種類などについては、担当の指導員に任せるといった役割分担が必要なのである。

さて、この指導員と生徒の比であるが、一人の指導員に１０人までが限界と思われる。きめ細かく指導するのなら７人ぐらいまでが理想であろう。どうしても、担当指導員が一人若しくは数人であり、一人の指導員に１０人より多くの人数を割り付かなければならないようならば、自由課題をあきらめ課題を統一した方がよい。

3 運用方法

さて、生徒には、課題申請書（提出書類１）を提出させ、それらの課題内容が決められた期間内で終了するものか、又は、それ以前にできあがるような簡単なものであるかなどの審査を行い、適正と思われものを承認する。この段階で、指導員間で話し合い、どの生徒を担当するかを決定する。

最初に詳細計画表（提出書類２）を提出させ、無理のない計画を立てさせる。計画表というとＰＥＲＴ図などが持ち出させるが、別にポラリス原潜を作成するような大規模なプロジェクトではないので、ふつうの計画表でよいと思われる。長くて６か月程度であるので、全体と詳細の二つを立てる必要もなく、１枚の詳細計画表のみで十分である。また、神奈川県では、２日を１単位とする単位制訓練を行っているので、２日の１つの仕事となるよう生徒に作業分解させ、計画を立てさせる。これで３か月ぐらいの計画を立てさせているが、特に問題はない。長期計画表と称してガントチャートも作らせた時期があったが、実際に作業に入ってから使用することがなく、作成しないことにした。

月、火曜日と木、金曜日が作業日であり、水曜日に火曜日までの作業内容について、担当の指導員から承認を受ける。また、金曜日までの作業は、翌月曜日の朝に同じように承認を受ける。承認を受けられない場合には、詳細計画表の「承認印」欄に印鑑が押されないため、

簡易の進捗状況表にもなっている。

月末に月間進捗状況報告書を提出させる。これにより、あるまとまった単位での作業の進行具合などを生徒自身が確認できる。また、このとき必要に応じて、先の詳細計画表を一部手直す場合もある。

最後には、マニュアルを含む全てのドキュメント類を完成させ、提出させる。

4 ソフトウェアの作成手順と期間

ソフトウェアのライフサイクルにおける各段階を踏まえ、生徒が作成するために必要な期間をみてみる。開発するものの規模にもよるので、これはある一例と考えていただきたい。

要求分析	2か月
システム外部設計	1か月
システム内部設計	1か月
プログラミング テスト	2か月（4GL等を用いれば1か月）

これから、実習期間を1か月、2か月、3か月と区切ることにより、それぞれあるまとまった実習が行えると思われる。

(1) 実習期間1か月

これは、4GLを備えたアプリケーションソフトウェア上でプログラム開発する程度であろう。こちらから、既に決定している内部設計書を渡し、プログラミング、デバッグでこのぐらいの日数を要する。ただし、数本のプログラムを作成するぐらいの規模である。

また、指導員が顧客となり、ニーズ調査を受け、またその手法を教授するのであれば、要求分析のみの課題も1か月を要する。3～4程度サブシステムで構成されるシステムの分析ぐらいが適当であろう。

(2) 自習期間2か月

COBOLやCといった汎用プログラミング言語を用いたプログラム開発をさせるのに良い期間である。この場合も、既に作成されている内部仕様書を提示して作業に取りかかることになる。

(3) 実習期間3か月

このくらいの期間になると自由課題とすることができる。システム設計に重点を置くのであれば、要求分析から外部仕様にいたるまで、プログラミング設計を主に行うのであれば、外部仕様、内部仕様とプログラミングにいたるまでである。ただし、汎用プログラミング言語を使用した場合は、十分にデバッグできない。4GLを用いればかなり質の高いシステムを作成できる。

(4) 実習期間6か月

これは、要求分析から、プログラミングまで行える期間である。このとき、要求分析は大変重要であり、必要があればその対象となるシステムを実際に運用している場所に取材に行かせることも必要である。

提出書類１

課 題 申 請 書

平成５年１２月１６日
情報処理系　２年　４０番　山本　太郎

次の内容のとおり、課題を作成したいと思いますので申請します。

課題名	ハードディスク管理ユーティリティ

開発目的：
　学校のパソコンのハードディスクの中身を見ていると、知らない人が勝手にファイルを作成したりして、オリジナルの状態を保てなくなっています。そこで、インストール時の状態をパックアップしておき、必要なときに必要なファイルのみを戻すことによって、ハードディスク内の復元化に費やす手間と時間を節約するものです。

概　　要：
　まず、始めにハードディスクを保存すべき状態にディレクトリ内を整理します。そして、本プログラムの「ユーティリティディスク」を用いて、ハードディスク全体、又は指定したディレクトリのみを、バックアップします。これらは、「診断ディスク」と「バックアップディスク」に分けられ、必要なときに管理者が「診断ディスク」を用いると登録されていないファイルの削除を行います。また、更新日付やファイル容量から書き換えられたとき予想されるファイルを「パックアップファイル」から戻します。

開発環境：　NEC　PC-98　model　８０　主記憶5.6MB　ハードディスク130MB
　　Borland　C++　Ver　3.1
　　MS-DOS　Ver　3.3D

動作環境：　MS-DOS　Ver　3.xx　以上が動作する　8086系マシン
　　主記憶　640KB以上

必要と思われる前提知識：

　MS-DOSのディレクトリ管理に関するシステムコール
　C言語によるソフトウェア割り込みの方法

担当指導員名	久　　保	承認印	

提出書類３

月間進捗状況報告書

平成６年１月３１日
情報処理系　２年　４０番　山本　太郎

１月分
当月に行った作業と進捗状況を報告します。

課題名	ハードディスク管理ユーティリティ

作業内容：
１　外部仕様の決定及び外部仕様書の作成
２　内部仕様の決定及び内部仕様書の作成

進捗状況：
１　外部仕様については決定し、仕様書の作成が完了している。
２　内部仕様については、ディレクトリに関するデータ構造を変更したため、モジュールの役割分担に多少時間がかかり、内部仕様書の作成を現在行っている。後１日あれば完成するものと思われる。

詳細計画表の変更の有無：

上記に報告した遅れは、２月２日（水）にて取り戻せるものと思われるので、特に変更の必要はない。

担当指導員名	久　保	承認印	

チェックシート１

要　求　仕　様　書

要求分析と現況把握

- 現状の作業体制が把握されているか。
- 顧客のニーズを把握しているか。
- 現状の作業内容の流れが理解されているか。
- 日常常識とされているものについても記述されているか・
- 矛盾点はないか。

提案システム

- 無駄な作業がないよう改善されているか。
- どの部分をコンピュータ化するか記述されているか。
- ファイル内のデータは、正規化されているか。

データフローダイアグラム

システムの内部と外部がしっかり定義されているか。

- システム内のものをエンティティとしていないか。
- システム外のものをシステム内に組み込んでいないか。

サブシステムに正しく分割されているか。

- サブシステムがまとまった機能を成しているか。
- １枚のデータフローダイアグラムに７つより多くのバブルがないか。
- １つのバブルに幾つもの機能を持たせていないか。
- 名前の内アローが存在しないか。

記入上のミスはないか。

- アローが交差されていないか。
- バブルに番号が順序たてて記入されているか。
- 違うデータに同じ名前をつけていないか。
- アローに動詞が記述されていないか。
- バブルは「～を～する」といった形式で記述されているか。
- アローに日付などの制御となるようなものはないか。

データディクショナリ

- データフローダイアグラムと矛盾なく定義されているか。
- 繰り返しなどの制御構造が、正しく記述されているか。
- 意味不明な用語が定義されていないか。

チェックシート１

内 部 仕 様 書

データ構造

- 必要なデータ構造が過不足なく記述されているか。
- 適正なコード設計が成されているか。
- 必要なファイルに適正な項目が定義されているか。

画面遷移図

- 画面の流れがよくわかるよう記述されているか。
- 機能キーが画面ごとに統一され記述されているか。

モジュール定義（Ｃ言語の場合）

- ソースファイルが機能ごとに適正に分割されているか。
- 変数のスコープ範囲が適正であるか。
- 内部のみで使用する変数を外部に公開していないか。
- リテラル値の取扱いが適正であるか。
- 機種に依存するものはまとめて定義されているか。

各関数定義（Ｃ言語の場合）

- 引き数の型、名称が定義されているか。
- 戻り値の型、役割が定義されているか。
- 必要な資源（ファイル、画面など）が定義されているか。
- 関数の機能と実現方法が記述されているか。
- 必要な関数が過不足なく記述されているか。

生徒が行う課題と指導員の役割について

生徒が行う段階	指導員の役割
要求分析	顧客としての要求提出、書類の記述方法のアドバイス
システム外部設計	良い操作性を持つシステムのアドバイス
システム内部設計	最良のデータ構造、モジュール分割の仕方のアドバイス
プログラミング	バグのでないプログラミングスタイルのアドバイス
テスト	より完全となるテスト項目の設定の仕方のアドバイス
運用	分かりやすいマニュアルの作成の仕方のアドバイス