長崎大学大学院生産科学研究科（博士前期課程）　　受験番号＿＿＿＿＿＿＿＿＿＿
電気情報工学専攻（情報システム工学系）
平成19年度入試問題
ソフトウェア

問１ 次の文は「問題の把握からプログラムの完成までのプログラミング過程」を５段階に分けて示したものである。下記の[ａ.]～[ｘ.]に当てはまる語句を記入せよ。

（ｉ）問題の把握と対策： 何が問題か。その問題を解決するにはどのようなシステムで対処するのかを考え，全体を見渡して人手や機械で処理し部分と，計算機で処理する部分とを明らかにする。

（ⅱ）プログラミングの設計： フローチャートを用いてシステムの開発工程を管理することを考えると，システムの設計，製造，テストなどの各開発工程によって，システム，プログラム，データの各フローチャートを準備することになる。この中のプログラムフローチャートはコンピュータに行わせる処理部分を取り出して，その手順を表したものである。その作成には必要なプログラムの決定と，プログラムの作成条件,および処理内容を細部に渡って記述した[ａ.] を作成する必要がある。次に，具体的な処理の手順を考え，既存のプログラムを利用する部分と，[ｂ.] 部分とを区別する。また，プログラムフローチャートとしては,[ｃ.]フローチャートとしてプログラム全体の処理の流れを捉え易いようにプログラム単位で概要を記したものと，[ｄ.] フローチャートとしてプログラミング言語で表現できるレベルまで詳細に処理内容を記したものを準備する。

（ⅲ）[ｅ.] ： プログラムの作成では，まずキーボードから入力する文字コードを[ｆ.]によって計算機内に取り込み，[ｇ.]プログラムを作成する。次に，[ｈ.]を用いて[ｇ.]プログラムを[ｉ.]の処理を通して[ｊ.]プログラムを作成する。さらに[ｊ.]プログラムに対して[ｋ.]を用いて[ｌ.]と結合させるリンク処理を行い,実行形式プログラムを作成する。実行形式プログラムに割り振る番地は,後で主記憶の任意の領域に配置できるよう，先頭番地を０とした[ｍ.]である。プログラムを実行すると，スーパーバイザが主記憶に領域を確保し，実行形式プログラムを主記憶に読み込んで[ｍ.]を[ｎ.]に変換し，実行開始番地へ制御を移して実行処理を行なう。

（ⅳ）実行と検査： 検査では[ｏ.] を用いてバグを検出したり，[ｐ.] の分かっているデータを用いて動作を確認する。計算機の内部ではこの段階の作業を[ｑ.] で実行している。(ⅲ)項のように複数の変換処理（手続き）を施して実行形式に変換するプログラム解釈方式を[ｒ.] 方式と呼ぶ。計算機のプログラム解釈方式には[ｒ.]方式以外にも[ｓ.] 方式がある。それは[ｔ.]言語で書かれたプログラムを，一文づつ解釈して実行するもので，[ｉ.]やリンクなどの処理を経ずに[ｕ.]という利点がある。そのため，[ｓ.] 方式は[ｒ.] 方式に比べ，[ｖ.]が容易な反面，実行速度が遅いと云う欠点がある 。

（ｖ）文書の作成：一度作ったプログラムは，故障対応、機能確認などの [ｗ.]が容易なように，作成に使用した各種文書を整理して保存しておく。また，他人でも使えるように，処理方法，データの入出力方法，使用方法，[ｘ.] などを文書にまとめ，残しておく。

【解答欄】

ａ.	ｂ.	ｃ.	ｄ.
ｅ.	ｆ.	ｇ.	ｈ.
ｉ.	ｊ.	ｋ.	ｌ.
ｍ.	ｎ.	ｏ.	ｐ.
ｑ.	ｒ.	ｓ.	ｔ.
ｕ.	ｖ.	ｗ.	ｘ.

長崎大学大学院生産科学研究科（博士前期課程）　　受験番号＿＿＿＿＿＿＿＿＿＿
電気情報工学専攻（情報システム工学系）
平成19年度入試問題
ソフトウェア

問２ 次の文中の[a.] ～[ｖ.]に，適切な語句，記号，数値，式，図を記入して答えよ。ただし，(1)項，(2)項は下記のように並んだ11個のデータに対しての問いに答えよ。

13，22，５，16，40，３，18，７，21，35，10

（１）線形探索で目的のデータを探索するための照合回数は，最小で[ａ．回]，最大で[ｂ．回]，平均で[ｃ．回]である。

（２）上記の 11 個のデータを降順にソートした後，そのデータを用いて完全２分探索木を描け[図：ｄ.]。このデータを２分探索した場合の照合回数は，最大で[ｅ．回]，平均で[ｆ．回]である。

（３）データの個数を n として，(1)項と(2)項の探索法の計算量を比較する。線形探索の時間計算量は最悪 2 分探索木の比較回数と同じになり，オーダー表記で[ｇ.]になる。一方，完全２分探索木は木の高さを h とし，葉の節点に探索したいデータがあるとすれば，最終的に葉で 1 つのデータを検出することになるので，木の高さに相当する回数，すなわち h 回の比較をすることになる。しかも n 個のデータは比較するごとに 2 分割されるので，データ n 個が h 回の分割後に 1 個のデータとなるので[式：ｈ.]が成り立つ。これを対数で表すと［式：ｉ.］となり，データの比較回数（時間計算量）は，オーダー表記で［ｊ.］となる。

（４）ハッシュ法による探索において，登録するキーデータをｘとし，ハッシュテーブルのサイズを S，ハッシュ関数をｈ1(ｘ)＝(ｘ) ｍｏｄ (S) とした場合，７個のキーｘ＝｛Ｃ，Ｄ，Ｆ，Ｊ，Ｌ，Ｎ，Ｔ｝はどのように格納されるか。ただし，S＝９とし，衝突に際しては「オープンアドレス法を用いて算出された番地 ｈ2(ｘ)=８－ｈ1(ｘ) に格納する」とするが，それでも衝突が起きた場合は，プラス方向の開番地とする。なお，テーブルの番地は 0 から始まるものとする。

このデータ格納状態でキー“Ｒ”の探索を行なうと，どんな経過をたどって探索が行なわれるかを記せ[ｌ.]。また，その時の探索結果は，何回の照合(データの比較)で得られるか[ｍ．回]。なお，キーデータの値は JIS コードの値とする（例 B=66)。

（５）ハッシュ法による探索においてデータ数がｎの場合の照合回数は，最大（最悪）で[ｎ．回]になる。しかし，一般にはハッシュテーブルがデータ数ｎよりも大きくなるように設計されるので，平均的には，ほぼ[ｏ．回]で探索できる。

（６）文字列探索においてテキスト文字列（ ＭＯＪＩＲＥＴＵＫＥＮＳＡ‥）を，照合文字列（ＵＫＥＮＳＡ）で探索する場合，文頭から照合文字列との一致を判定し，一致しなければ，1 文字づつずらして探索すると，[ｐ．]の照合で一致が検出でき，アドレス７が検出できる（アドレスは０番地から始まるものとし，照合は文字単位とする)。

簡易ボイヤー・ムーア法（ＢＭ法）で探索する場合は，最初に[ｑ．]を照合し，一致していれば文字ごとの一致を判定する。[ｑ．]が一致しなければ，照合した文字が照合文字列の[ｒ．]であれば１文字ずらし，[ｓ．]であれば２文字ずらし，A, E, K, N, S, U以外であれば[ｔ．]文字ずらして次の照合を行なう。その結果，ＢＭ法では[ｕ.回]の照合で照合文字列の存在位置を検出できる（ ここでの[ｒ．][ｓ．]は，前から，あるいは後からも付して記入せよ ）。なお，ｕを求める際の文字の照合過程の説明を[ｖ.]の枠内に記入せよ。

【解答欄】は別紙に示す。

長崎大学大学院生産科学研究科（博士前期課程）　　受験番号＿＿＿＿＿＿＿＿＿＿
電気情報工学専攻（情報システム工学系）
平成19年度入試問題
ソフトウェア

問２の【解答欄】

a． 回						b． 回				図 d． ○
c． 回										
e． 回						f． 回				
g．						式 h.				
式 i．						j．				
k．	0	1	2	3	4	5	6	7	8	
文 l．										
m． 回						n． 回				o． 回
p．						q．				r．
t．						u．				
文 v．										

長崎大学大学院生産科学研究科 (博士前期課程) **受験番号 _____**
電気情報工学専攻 (情報システム工学系)
平成 19 年度 入学試験問題
ソフトウェア

問 3 基本ソフトウェアに関する以下の文章中の空欄 (1〜20) を埋めよ。なお [n.] で表した空欄に関してはもっとも適切な語句を示し (語句の一部が既に記入されているものもある)、{ n. a:…, b:…, c:… } で表した空欄に関してはその中の選択肢 (a〜c) から適切な語句を選ぶこと (解答欄には a,b,c ではなく語句そのものを記述すること)。

オペレーティングシステムは計算資源の管理を一手に行う管理プログラムである。従ってクロックの現在時刻を調整する、ファイルを生成する、ヒープメモリを確保するといった一般プログラム中の資源利用は全て OS に代行してもらうためのリクエストを出さなければならない。そのために OS は関数のようなインターフェイスを提供している。これを [**1. コール**] と言う (アプリケーションプログラムに対するインターフェイスであるのでこのようなものを一般的に [**2.**] (3 文字の英単語) ということもある)。

[**1. コール**] は関数呼び出しに似ているが呼び出しの実現手段に違いがある。例えば C 言語のような高級 (高水準) 言語で書かれたプログラムは [**3.**] を用いて機械語の実行ファイルに変換されるが、元プログラム中に出現する関数呼び出し構文は呼び出し先関数の先頭番地へのサブルーチンコールをする命令 (例えば CALL 命令) に変換されるのに対し、[**1. コール**] は [**4.**] を起こす命令 (例えば i386 では INT 命令) へと変換される。これはリクエストを処理する OS 内部の関数 (ハンドラ) は、[**4.**] を起こすプロセスとは別の [**5. 空間**] に含まれるため、通常の番地指定では指すことができないからである。

ここでプログラミング言語処理系から見た機能単位の呼び出しコストの比較をしてみる。関数呼び出しの場合はその引数をプロセスの [**6. 領域**] 内に関数呼び出し側で push する、あるいは CPU 内の [**7.**] に格納することが必要であるため、単純な goto 文 (機械語レベルでは JUMP 命令に相当) よりも呼び出しの手間が必要であるが、[**4.**] の場合は、それに加えて、

- CPU のモードを一般プログラム用の [**8. モード**] から全ての命令が使える [**9. モード**] に変更すること
- 中断したプロセスを後で再開するために、そのプロセスのプログラムカウンタや全ての [7.] などの情報をプロセスの情報を保持する領域である [**10.**](3 文字の英単語) に保存すること
- 処理終了後には OS 内で [**11.**] がどのプロセスを再開させるかを決定する必要があること

などから、(例え処理内容が同一であるとしても) 関数呼び出しよりも [**1. コール**] はその呼び出しにずっと時間が掛かることになる。

また、オブジェクト指向言語では [**12.**] の呼び出しが手続き言語での関数呼び出しに対応するが、関数と違って [**12.**] には同名のものが複数存在する可能性があり、そのうちのどれを選択すべきかはコンパイル時ではなく実行時にしか決まらないことがある (これを [**13. 性**] といいオブジェクト指向言語の特徴の一つとされている)。従って機械語へのコンパイル時にジャンプ先番地が決定された CALL 命令に置き換えることができないため、[**12.**] 呼び出しも通常の関数呼び出しよりも一般に呼び出しコストが大きい。代わりに、OS はセキュリティを、オブジェクト指向言語はプログラム開発での自由度を手に入れることができた。

長崎大学大学院生産科学研究科 (博士前期課程) **受験番号 _____**
電気情報工学専攻 (情報システム工学系)
平成 19 年度 入学試験問題
ソフトウェア

なお、[**4.**] の発生は [**1. コール**] によるものだけではない。[**14.**] を定期的に切替えるためにタイマーを使っている多くの OS では少なくともおおよそ **{ 15. a:1 ミリ 秒、b:1 マイクロ秒、c:1 秒 }** 程度の間隔で常に [**4.**] を処理することになる。このタイマーを使った手法のように強制的に [**14.**] を切替えるスケジューリングを **{ 16: a:ノンプリエンプティブ, b:プリエンプティブ }** スケジューリングと言う。

仮想記憶を用いている OS においては、もしも、切替えた [**14.**] が **{17. a:レジスタ、b:キャッシュ、 c:主記憶 }** 上に存在しないページを参照した場合は一般に
{18. a:ページアウト, b:ページイン, c:ページフォールト } →
{19. a:ページアウト, b:ページイン, c:ページフォールト } →
{20. a:ページアウト, b:ページイン, c:ページフォールト }
という順番で対応することになる。

解答欄

1. コール	2.	3.	4.
5. 空間	6. 領域	7.	8. モード
9. モード	10.	11.	12.
13. 性	14.	15.	16.
17.	18.	19.	20.

長崎大学大学院生産科学研究科 (博士前期課程) **受験番号 _____**
電気情報工学専攻 (情報システム工学系)
平成 19 年度 入学試験問題
ソフトウェア

問 4 以下の文章中の空欄 (1〜10) を埋めよ。なお [*n.*] で表した空欄に関してはもっとも適切な語句を示し (語句の一部が既に記入されているものもある)、 { *n.* a:…, b:…, c:…} で表した空欄に関してはその中の選択肢 (a〜c) から適切な語句を選ぶこと (解答欄には a,b,c ではなく語句そのものを記述すること)。

以下はある言語での関数または変数の宣言文の文法 (<Declaration>) を [**1.**] (3 文字の英単語) 記法で表したものである (正確には、以下ではその拡張された記法を用いている)。なお [**2. 記号**] は<>で、[**3. 記号**] はシングルクォートで囲んで表す。[ ] はグループの区切りとして用いる。

```
<Declaration> ::= <FunctionDeclaration> | <VariableDeclaration>

<FunctionDeclaration> ::= 'def' <type> <id> '('  <argList> ')' ';'
<VariableDeclaration> ::= [ <type> <id> ';' ]   |   [ <type> <id> '[' <pnum> ']' ';' ]

<type>    ::= [ 'class' <id> ]  | 'int' | 'float'
<argList> ::= [ <type> <id> ',' ]*

<id>      ::= <id>['a'|'b'|…|'z']   |   ['a'|'b'|…|'z']
<pnum>    ::= [ '+' ['0'|'1'|…|'9']+ ]   |   ['0'|'1'|…|'9']+
```

(この中の下の 2 つのルールに関しては記号間には空白を挟まないものとする。)

これらの文法の基では (意味については考えないとして)、以下の「変数宣言文」1〜6:

1. `int a1;`
2. `double a;`
3. `class ab;`
4. `float ab[];`
5. `class ab ba[00];`
6. `int ab[3+1];`

の中では、[**4.**] (番号で解答、複数ある場合は全て書くこと) が正しい構文となる。

関数宣言に関しては、以下の「関数宣言文」1〜6:

1. `def int ab();`
2. `def int ab(,);`
3. `def int ab(int a);`
4. `def int ab(int a, );`
5. `def int ab(int a, int b);`
6. `def int ab(int a, int b,);`

の中では、[**5.**] (番号で解答、複数ある場合は全て書くこと) が正しい構文となる。

なお、この構文から<FunctionDeclaration>中の [**6.**] を取り除くとあいまい性が生じ、再帰的 **{ 7. a:上, b:下, c:右, d:左 }** 向き構文解析の代表である LL(1) 文法の構文器では対応できない。また、<id>には **{ 8. a:上, b:下, c:右, d:左 }** 再帰性の問題が出現しているので文法の書き換えが必要である。

この文法では <id> がいくらでも長い英字列を受け付けるので、変数名関数名として使える名前が有限でないほどに多く、<Declaration> の言語 (その文法で正しいと判断される文の集合、すなわち<Declaration>にマッチするものの集合) は無限集合となる。もし<id> の定義を

```
<id> ::= 'a' | 'b'
```

と変更するなら (そしてそれ以外は変更しないなら)、<id> の言語は **{ 9. a:有限集合, b:無限集合 }** となり、一方、<Declaration>の言語は **{ 10. a:有限集合、b:無限集合 }** となる。

解答欄

1.	2. 記号	3. 記号	4.
5.	6.	7.	8.
9.	10.		

長崎大学大学院生産科学研究科（博士前期課程）　　受験番号＿＿＿＿＿＿＿＿
電気情報工学専攻（情報システム工学系）
平成19年度入試問題
ソフトウェア

問５　以下の問いについて答えよ。

（１） 次の仕様の関数を作成せよ。なお，言語は C である。

関数名		rotation		関数の型	void	戻り値	無し
引数 (*)	入力	整数型	x	回転前の点の x 座標値をセットする。			
		整数型	y	回転前の点の y 座標値をセットする。			
		実数型	rad	回転する角度をラジアンでセットする。（範囲は　0～2π である。）			
	出力	整数型ポインタ	ptx	回転後の点の x 座標値がセットされる。			
		整数型ポインタ	pty	回転後の点の y 座標値がセットされる。			
機能 (**)		引数として入力された，座標値(x, y)に対して原点を中心に角度 rad(ラジアン) 回転した座標値(x', y')を求め，引数の出力であるポインタ変数 ptx，pty にセットする。なお，2 次元座標系で 1 点(x, y)が原点を中心に角度θ回転する場合，移動した点(x', y')は次式によって求められる。 $\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$					

（＊）引数については，構造体を使用してもよい。

（＊＊）以下の仕様の sin, cos 関数を使用すること。

```
#include <math.h>
        :
double sin(double x);
double cos(double x);
```

引数 x に角度（ラジアン）を指定する。戻り値は，角度 x での sin 関数または cos 関数の値である。指定できる角度は０～πであり，それ以外の角度を指定した場合の結果は保証されない。

（＊＊＊）(3)と同じ解答欄に解答すること。

（２）(1)の関数 rotation の動作を確認するため，関数内のすべての命令文を実行するテストデータおよびその際の出力を，下表の空欄に記入してチェックリストを完成させよ。

#	入　力	出　力	結果
1			記入不要

（３）(1)の関数を(2)のテストデータで実行し，その結果を確認するための関数 main を作成せよ。なお，確認は人間が目視で行うものとし，得られた結果は printf 関数などを用いて画面上に表示すればよい。

（１），（３）の解答欄

```
#include <stdio.h>
#include <math.h>
#define   PI   3.1415
```

長崎大学大学院生産科学研究科（博士前期課程）　　受験番号＿＿＿＿＿＿＿＿
電気情報工学専攻（情報システム工学系）
平成19年度入試問題
ソフトウェア

問６　以下の文章の空欄［1.］～［10.］を該当する語句で埋めよ。なお，同じ用語を何度使用しても良い。

（１）　1960 年代に，［１.］文が多用されたプログラムによって，プログラムの可読性の低下やプログラムの誤りなどが問題となり，構造化プログラミングが提唱されました。構造化プログラミングの原理の一つは，［２.］，［３.］，［４.］の３つの組み合わせですべてのプログラムを作成できるというものです。もう一つは，複雑な問題を一度にプログラムとして記述するのではなく，その問題全体の概要をとらえて，徐々にいくつかの簡単な部分に分割し詳細化していく，［５.］の原理です。したがって，構造化プログラミングでは，いくつかの簡単なプログラムモジュールの集まりとしてプログラムが作成されます。このことにより，プログラムの可読性の向上，プログラムを簡単な部分から作成することによるプログラムの誤りの減少などの効果が期待できます。

（２）　C 言語では，定義した変数が有効な範囲をスコープと言います。変数をスコープという観点から分類すると［６.］変数と［７.］変数に分類できます。［６.］変数は関数内で定義し，有効範囲はその［８.］内です。［７.］変数は関数外で定義し，有効範囲はその［９.］内です。［７.］変数を定義した［９.］以外で利用したい場合は，その変数を［１０.］文で外部変数として宣言します。

異なる関数間では，互いの関数内で定義した［６.］変数を共用することはできません。この機能により，変数のスコープを関数内に制限することができ，プログラミングにおける変数の取扱いミスを防ぐことができます。もしも，複数の関数でデータの受け渡しをするような場合は，引数を用いるなどして，関数間で共有する変数を使用しない方がプログラミングにおける誤りを減らすことができます。

しかし，関数間で共有変数を使用しなければならない場合もあります。［７.］変数は関数間で変数を共有するような場合に使用します。ただし，［６.］変数と［７.］変数を同じ名前で定義した場合，［６.］変数を定義した関数内では，［６.］変数が有効となり，［７.］変数は隠蔽されます。［７.］変数を使用する場合は，その変数名を一見して［７.］変数と分かるような名前にするなどの工夫をすれば，プログラミングの際の誤りを減らせるでしょう。

解答欄

１.	２.	３.	４.	５.
６.	７.	８.	９.	１０.