«به نام خدا»

درس سیستم‌های نهفته و بی‌درنگ

ترم بهار سال ۱۴۰۱-۱۴۰۲

پاسخنامه تمرین تئوری پایانی- از فصل ۱ تا ۱۳

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

سوال ۱:

درست یا نادرست بودن جملات زیر را مشخص کنید و در صورت نادرست بودن، صورت درست آن را بنویسید.

۱) یک زمانبندی feasible است اگر یکی نیازمندی‌های داده شده را برآورده کند.

نادرست، یک زمانبندی feasible است اگر تمام نیازمندی‌های داده شده را برآورده کند.

۲) دو تا از ویژگی‌های الگوریتم FIFO این است که سریع است و قابلیت زمانبندی پایینی دارد.

درست.

۳) اگر یک الگوریتم نتواند یک زمانبندی feasible برای یک task set انجام دهد، آنگاه آن task set، feasible نخواهد بود.

نادرست، اگر یک الگوریتم بتواند یک زمانبندی feasible برای یک task set انجام دهد، آنگاه آن task set، feasible خواهد بود.

۴) اگر یک الگوریتم از نظر feasibility، optimal باشد آنگاه این الگوریتم مقدار maximum lateness را minimize خواهد کرد.

نادرست، اگر یک الگوریتم مقدار maximum lateness را minimize خواهد کرد آنگاه آن الگوریتم از نظر feasibility، optimal است. (برعکس آن درست نیست)

۵) الگوریتم SRJF (Shortest Remaining Job First) مقدار میانگین response time را مینیمم می‌کند.

نادرست، SJF که non-preemptive است مقدار میانگین response time را مینیمم می‌کند.

۶) الگوریتم table-driven به شدت نسبت به بهینه سازی‌های زمانبندی انعطاف پذیر است.

درست.

۷) الگوریتم EDF می‌تواند هر نوع task set که feasible است را زمانبندی کند.

نادرست، الگوریتم EDF فقط می‌تواند Task setهایی را که preemptive و Feasible هستند، زمانبندی کند.

۸) پیاده سازی‌هایی که مقدار زمان اجرایی آن‌ها متفاوت است می‌توانند پیچیدگی زمانی یکسانی داشته باشند.

درست.

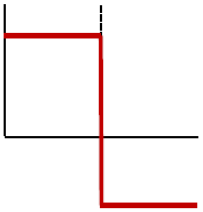
سوال ۲:

برای هر یک از سیستم‌های زیر مشخص کنید که چه نوع deadline دارند (Hard, Firm, Soft) و نمودار بهره‌وری آن‌ها را بصورت حدودی رسم نمایید و برای الگوریتم‌های پیشنهادی ،با توجه به سیستمی که در آن اجرا می‌شوند، مشخص کنید که کدام یک از ویژگی‌های زیر را دارند.

(preemptive / non-preemptive , work-conserving / non-work-conserving , static / dynamic , offline / online)

۱) Nuclear reactor control: fix priority

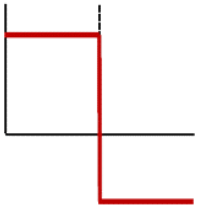
Hard



Preemptive, online, static, non-work-conserving

۲) Air traffic control systems: cycling scheduling

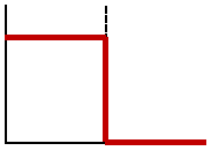
Hard



Preemptive, non-work-conserving, static, offline

۳) Mobile phone: Round Robin

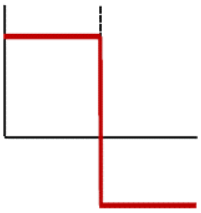
Firm



Preemptive, non-work-conserving, dynamic, online

۴) Flight control: EDF

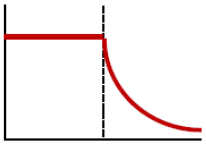
Hard



preemptive, non-work-conserving, dynamic, online

۵) Ticket sales system: FIFO

Soft



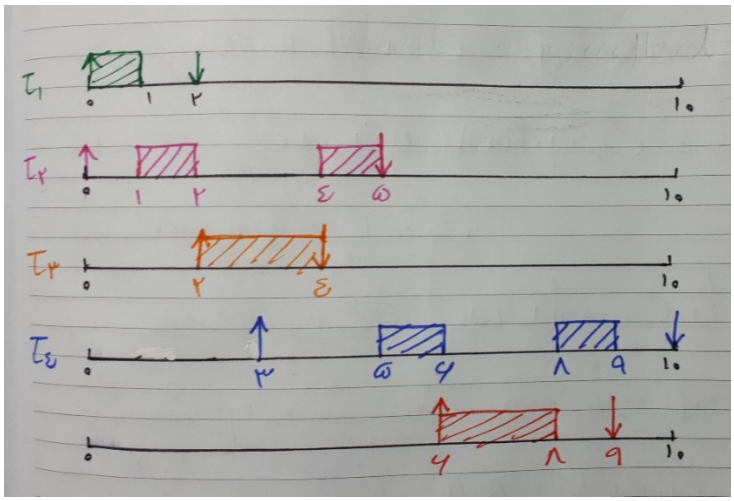
Non-preemptive, non-work-conserving, dynamic, online

سوال ۳:

با توجه به task set داده شده به سوالات زیر پاسخ دهید.

	ai	WCETi	Di
τ1	0	1	2
τ2	0	2	5
τ3	2	2	2
τ4	3	2	7
τ5	6	2	3

۱) این task set را توسط الگوریتم EDF زمانبندی کنید و شکل آن را رسم کنید.



۲) جدول زیر را با توجه به زمانبندی انجام شده در قسمت قبل پر کنید.

	Offset Time	Start Time	Finish Time	Response Time	Slack Time	Lateness Time
τ1	0	0	1	1	1	-1
τ2	0	1	5	5	0	0
τ3	2	2	4	2	0	0
τ4	3	5	9	6	1	-1
τ5	6	6	8	2	1	-1

۳) ماکسیمم تعداد deadlineهایی که امکان دارد توسط هر نوع الگوریتم زمانبندی miss شوند، چقدر است؟ چگونه؟

برای هر ۵ تا از task ها امکان miss شدن وجود دارد. به گونه‌ای که می‌توانیم الگوریتمی طراحی کنیم که برای مثال در ابتدا ۱۰ Clock صبر کند و بعدا شروع به کار کند که در این صورت تمام deadline ها قبل از شروع miss خواهند شد.

۴) در سومین Clock، هر کدام از process ها در کدام یک از وضعیت‌های زیر هستند؟

(New, Ready, Running, Waiting, Terminated)

Terminated	τ_1
Waiting	τ_2
Running	τ_3
Ready	τ_4
New	τ_5

سوال ۴:

با توجه به task set داده شده به سوالات زیر پاسخ دهید.

	a_i	$WCET_i$	D_i	T_i
τ_1	0	1	10	10
τ_2	0	2	5	5
τ_3	0	3	15	15

۱) آیا این task set، feasible است؟ دلیل خود را بیان کنید. (بدون انجام زمانبندی به این سوال پاسخ دهید)

بله، چونکه $U = 0.7 \leq 1$ است بنابراین این مجموعه وظایف feasible است.

$$U = 0.1 + 0.4 + 0.2 = 0.7$$

۲) آیا این task set، توسط یکی از انواع الگوریتم‌های fix priority قابل زمانبندی است؟ دلیل خود را بیان کنید. (بدون انجام زمانبندی به این سوال پاسخ دهید)

الگوریتم RM در زمانی که T = D بود، یک الگوریتم Optimal از نظر feasibility برای الگوریتم‌های Fix priority بود. بنابراین از تست hyperbolic bound استفاده می‌شود:

$$\prod_{i=1}^{n}(U_i + 1) \leq 2$$

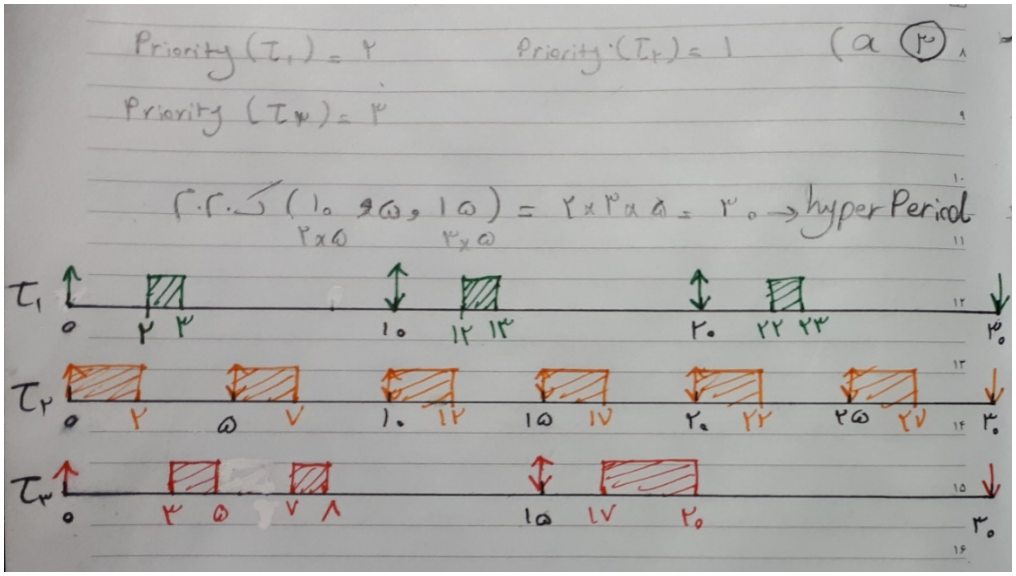
$$(0.1 + 1)(0.4 + 1)(0.2 + 1) = 1.848 \leq 2$$

چونکه این شرط برقرار است یعنی این task set توسط الگوریتم RM قابل زمانبندی است.

بنابراین با توجه به قضیه‌ی گفته شده در بالا، این task set چونکه R=D است توسط تمام الگوریتم‌های fix priority قابل زمانبندی است.

۳) در صورتی که قسمت ۲ جوابش مثبت است به قسمت a پاسخ دهید و در صورتی که جواب آن منفی است به قسمت b پاسخ دهید (تنها باید یکی از دو قسمت زیر حل شود).

الف) این task set را توسط الگوریتم DM زمانبندی کنید و با رسم شکل نشان دهید.



سوال ۵:

	C_i	T_i
τ_1	10	20
τ_2	3	10
τ_3	1	5

۱) تعیین کنید که آیا این سیستم ردیابی بر اساس **آزمون‌های** RM schedulability قابل زمانبندی است یا خیر. (هر سه آزمون)

برای تشخیص قابل زمانبندی بودن دستگاه ردیابی با استفاده از **آزمون‌های** زمانبندی RM نیازمند انجام آزمون‌های زیر هستیم.

۱) آزمون زمانبندی EDF:

طبق آزمون زمانبندی EDF، مجموع استفاده‌ی تسک‌ها از CPU باید کمتر یا مساوی ۱ باشد.

$$U = (C1/T1) + (C2/T2) + (C3/T3)$$

$$U = (10/20) + (3/10) + (1/5)$$

$$U = 0.5 + 0.3 + 0.2$$

$$U = 1$$

مجموع استفاده CPU برابر ۱ است که کمتر یا مساوی ۱ است. بنابراین، این مجموعه وظایف براساس آزمون EDF قابل زمانبندی است.

۲) آزمون L&L:

آزمون استفاده مبتنی بر L&L شرایط لازم و کافی برای زمانبندی با استفاده از الگوریتم RM را ارائه می‌دهد. این آزمون بیان می‌کند که اگر استفاده کلی CPU (U) کمتر یا مساوی $U \leq n(2^{\frac{1}{n}} - 1)$ باشد، که در آن n تعداد وظایف است، آنگاه سری وظایف قابل زممانبندی است.

$$U = 1$$

$$\text{Bound} = 3\left(2^{\frac{1}{3}} - 1\right) \approx 0.7798 \not\geq 1$$

از آنجایی که U = 1 بزرگتر از مقدار محدوده (۰.۷۷۹۸) است، مجموعه وظایف براساس آزمون L&L قابل زمانبندی نیست.

۳) آزمون Hyperbolic Bound:

مرز هایپربولیک یک آزمون دیگر است که به طور خاص برای الگوریتم RM استفاده می‌شود. این آزمون مرز بالایی را برای استفاده کلی CPU در یک مجموعه وظایف قابل زمانبندی فراهم می‌کند.

$$\prod_{i=1}^{n}(U_i + 1) \leq 2$$

$$(0.5 + 1)(0.3 + 1)(0.2 + 1) = 2.34 \not\leq 2$$

در این مورد،۲.۳۴ بزرگتر از مقدار مرز ۲ است، بنابراین این مجموعه وظایف براساس آزمون مرز هایپربولیک قابل برنامه‌ریزی نیست.

در نتیجه با توجه به آزمون‌های فوق، نتوانستیم اثباتی برای زمانبندی این مجموعه توسط الگوریتم RM پیدا کنیم ولی در همین حال نمی‌توان نتیجه گرفت که این مجموعه قابل زمانبندی نیست.

۲) صحت آزمون‌های (۱) را بر اساس **تحلیل‌های** RM schedulability بررسی کنید. ستون اولویت را در صورت نیاز پر کنید. یافته‌های خود را بیان کرده و با نتایج آزمون‌های (۱) مقایسه کنید. (هر سه تحلیل)

۱) تحلیل RTA:

ابتدا باید این مجموعه وظایف را بر اساس Rate اولویت بندی کنیم.

	C_i	T_i	P_i	R_i
τ_1	10	20	3	20
τ_2	3	10	2	4
τ_3	1	5	1	1

1) $R_3^{(0)} = 1$

$R_3^{(1)} = 1 \leq 1 \; stop$

$1 \leq 5$ ✓

2) $R_2^{(0)} = 3$

$R_2^{(1)} = 3 + \left\lceil \frac{3}{5} \right\rceil \times 1 = 4$

$R_2^{(2)} = 3 + \left\lceil \frac{4}{5} \right\rceil \times 1 = 4 \leq 4 \; stop$

$4 \leq 10$ ✓

3) $R_3^{(0)} = 10$

$R_3^{(1)} = 10 + \left\lceil \frac{10}{5} \right\rceil \times 1 + \left\lceil \frac{10}{10} \right\rceil \times 3 = 10 + 2 + 3 = 15$

$R_3^{(2)} = 10 + \left\lceil \frac{15}{5} \right\rceil \times 1 + \left\lceil \frac{15}{10} \right\rceil \times 3 = 10 + 3 + 6 = 19$

$R_3^{(3)} = 10 + \left\lceil \frac{19}{5} \right\rceil \times 1 + \left\lceil \frac{19}{10} \right\rceil \times 3 = 10 + 4 + 6 = 20$

$R_3^{(4)} = 10 + \left\lceil \frac{20}{5} \right\rceil \times 1 + \left\lceil \frac{20}{10} \right\rceil \times 3 = 10 + 4 + 6 = 20 \leq 20 \; stop$

$20 \leq 20$ ✓

بر خلاف نتایج آزمون‌ها زمانبندی RM، مشاهده می‌کنیم که بر اساس تحلیل RTA این مجموعه وظایف قابل زمانبندی است. توجه کنید که تحلیل RTA لازم و کافی است که یعنی اگر $R_i > D_i$ بود، می‌توانستیم با قطعیت نتیجه بگیریم که این مجموعه وظایف توسط الگوریتم RM قابل زمانبندی نیست.

۲) Park test:

1) $1 \leq 5$

2) $3 + \left\lceil \frac{10}{5} \right\rceil \times 1 = 5 \leq 10$

3) $10 + \left\lceil \frac{20}{5} \right\rceil \times 1 + \left\lceil \frac{20}{10} \right\rceil \times 3 = 10 + 4 + 6 = 20 \leq 20$

چونکه تمام وظایف شرط این آزمون را برآورده کرده‌اند، بنابراین این مجموعه وظایف با الگوریتم RM قابل زمانبندی است.

۳) **harmonic tasks test**:

اگر مجموعه وظایفی هارمونیک باشد شرط $U \leq 1$ برای زمانبندی RM لازم و کافی است. بنابراین چونکه مجموعه وظایف داده شده هارمونیک است و $U = 1 \leq 1$ می‌توانیم بدون انجام هیچ یک از آزمون‌های قسمت اول و یا تحلیل‌های دو بخش قبلی نتیجه‌گیری کنیم که این مجموعه وظایف قابل زمانبندی توسط الگوریتم RM است.

۳) فرض کنید که زمانبندی RM در جدول بالا باید به زمانبندی EDF با arrival timeهای زیر تبدیل شود.

$a_1 = 1, a_2 = 3, a_3 = 0$

(الف) آزمون زمانبندی EDF

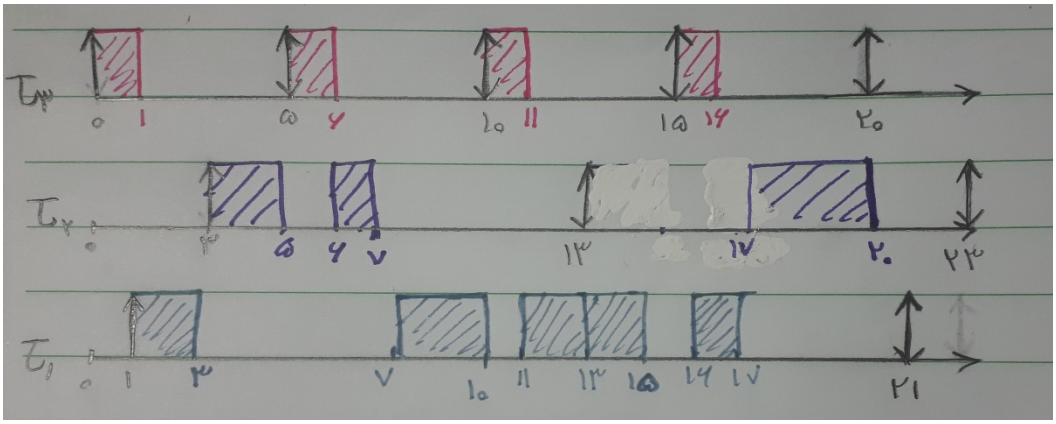
این مجموعه وظایف بر اساس آزمون EDF که در ادامه آمده است، قابل زمانبندی با الگوریتم EDF است.

$U = (10/20) + (3/10) + (1/5)$

$U = 0.5 + 0.3 + 0.2$

$U = 1 \leq 1$

(ب) روش نمودار زمانی



۴) به صورت مختصر تفاوت‌های بین زمانبندی‌های زیر را توضیح دهید: (هر کدام سه مورد)

(در این سوال به هر سه موردی که تفاوت‌ها را بیان کند نمره داده می‌شود.)

الف) RM و DM

RM (Rate Monotonic Scheduling) و DM (Deadline Monotonic Scheduling) هر دو الگوریتم‌های زمانبندی بر اساس اولویت هستند.

۱) درRM وظایف براساس دوره‌هایشان اولویت‌بندی می‌شوند (وظایف با دوره‌ی کوتاه‌تر اولویت بالاتری خواهند داشت)، در حالی که در DM وظایف براساس ددلاین‌های نسبی اولویت بندی می‌شوند (وظایف با ددلاین کوتاه‌تر اولویت بالاتری خواهند داشت).

۲) RM فرض می‌کند که ددلاین‌های نسبی برابر با دوره‌های وظایف هستند، در حالی که DM مهلت‌هایی را برای هر وظیفه تعریف می‌کند.

۳) RM تضمین می‌کند که یک مجموعه وظایف قابل زمانبندی است اگر استفاده کل CPU در محدوده خاصی باشد. DM می‌تواند محدودیت‌های بهتری برای استفاده از CPU برای برخی از مجموعه وظایف در مقایسه با RM ارائه دهد.

ب) RM و EDF

RM (Rate Monotonic Scheduling) و EDF (Earliest Deadline First) هر دو الگوریتم‌های زمانبندی بر اساس اولویت هستند.

۱) در RM وظایف براساس دوره‌هایشان اولویت‌بندی می‌شوند، در حالی که در EDF وظایف براساس ددلاین‌های مطلق خودشان اولویت‌بندی می‌شوند.

۲) RM فرض می‌کند که ددلاین‌های نسبی برابر با دوره‌های وظایف هستند، در حالی که EDF برای هر وظیفه ددلاین‌های ضروری را در نظر می‌گیرد.

۳) یک مجموعه وظایف قابل زمانبندی است اگر استفاده کل CPU در محدوده خاصی باشد. EDF تضمین می‌کند که استفاده کل CPU اگر کمتر یا مساوی ۱ باشد، آن مجموعه وظایف قابل زمانبندی است.

سوال ۶:

فرض کنید مجموعه‌ای از وظایف دوره‌ای با وظایف مستقل زیر وجود دارد.

	C_i	T_i	D_i
τ_1	20	100	100
τ_2	30	145	145
τ_3	68	150	150

۱) برای این مجموعه وظایف هر سه آزمون **utilization-based** را انجام دهید.

۱) آزمون زمانبندی EDF:

طبق آزمون زمانبندی EDF، مجموع استفاده‌ی تسک‌ها از CPU باید کمتر یا مساوی ۱ باشد.

$$U = (C1/T1) + (C2/T2) + (C3/T3)$$

$$U = (20/100) + (30/145) + (68/150)$$

$$U = 0.2 + 0.2069 + 0.4533 \approx 0.8602$$

مجموع استفاده CPU برابر ۰.۸۶۰۲ است که کمتر یا مساوی ۱ است. بنابراین، این مجموعه وظایف براساس آزمون EDF قابل زمانبندی است.

۲) آزمون L&L:

آزمون استفاده مبتنی بر L&L شرایط لازم و کافی برای زمانبندی با استفاده از الگوریتم RM را ارائه می‌دهد. این آزمون بیان می‌کند که اگر استفاده کلی CPU (U) کمتر یا مساوی $U \leq n(2^{\frac{1}{n}} - 1)$ باشد، که در آن n تعداد وظایف است، آنگاه سری وظایف قابل زمانبندی است.

$$U = 0.8602$$

$$\text{Bound} = 3\left(2^{\frac{1}{3}} - 1\right) \approx 0.7798 \not\geq 0.8602$$

از آنجایی که U = 0.8602 بزرگتر از مقدار محدوده (۰.۷۷۹۸) است، مجموعه وظایف براساس آزمون L&L قابل زمانبندی نیست.

۳) آزمون Hyperbolic Bound:

مرز هایپربولیک یک آزمون دیگر است که به طور خاص برای الگوریتم RM استفاده می‌شود. این آزمون مرز بالایی را برای استفاده کلی CPU در یک مجموعه وظایف قابل زمانبندی فراهم می‌کند.

$$\prod_{i=1}^{n}(U_i + 1) \leq 2$$

$$(0.4533 + 1)(0.2069 + 1)(0.2 + 1) \approx 2.1048 \not\leq 2$$

در این مورد،۲.۱۰۴۸ بزرگتر از مقدار مرز ۲ است، بنابراین این مجموعه وظایف براساس آزمون مرز هایپربولیک قابل برنامه‌ریزی نیست.

در نتیجه با توجه به آزمون‌های فوق، نتوانستیم اثباتی برای زمانبندی این مجموعه توسط الگوریتم RM پیدا کنیم ولی در همین حال نمی‌توان نتیجه گرفت که این مجموعه قابل زمانبندی نیست.

۴) فرض کنید زمان ورود اولیه تمام وظایف t = 0 باشد.

الف) با استفاده از زمان‌بندی RM، آیا هر سه ددلاین رعایت خواهند شد؟ دلیل خود را بیان کنید.

راه حل اول:

از آنجایی که نتوانستیم نتیجه‌ای از آزمون‌های زمانبندی RM بگیریم بنابراین باید از تحلیل‌های RM استفاده کنیم.

۱) این مجموعه وظایف هارمونیک نیست بنابراین نمی‌توانیم به جوابی از این طریق برسیم.

۲) park test:

$$20 \leq 100$$
$$30 + \left\lceil \frac{145}{100} \right\rceil \times 20 = 30 + 40 = 70 \leq 145$$
$$68 + \left\lceil \frac{150}{100} \right\rceil \times 20 + \left\lceil \frac{150}{145} \right\rceil \times 30 = 68 + 40 + 60 = 168 \not\leq 150$$

چونکه وظیفه سوم شرط این آزمون را برآورده نکرد، بنابراین نمی‌توانیم نتیجه‌ای از این آزمون بگیریم.

۳) RTA:

راه حل اول:

1) $R_1^{(0)} = 20$

$R_1^{(1)} = 20 \leq 20\ stop$

$20 \leq 100$ ✓

2) $R_2^{(0)} = 30$

$R_2^{(1)} = 30 + \left\lceil \frac{30}{100} \right\rceil \times 20 = 50$

$R_2^{(2)} = 30 + \left\lceil \frac{50}{100} \right\rceil \times 20 = \ 50 \leq 50\ stop$

$50 \leq 145$ ✓

3) $R_3^{(0)} = 68$

$R_3^{(1)} = 68 + \left\lceil \frac{68}{100} \right\rceil \times 20 + \left\lceil \frac{68}{145} \right\rceil \times 30 = 68 + 20 + 30 = 118$

$R_3^{(2)} = 68 + \left\lceil \frac{118}{100} \right\rceil \times 20 + \left\lceil \frac{118}{145} \right\rceil \times 30 = 68 + 40 + 30 = 138$

$R_3^{(3)} = 68 + \left\lceil \frac{138}{100} \right\rceil \times 20 + \left\lceil \frac{138}{145} \right\rceil \times 30 = 68 + 40 + 30 = 138 \leq 138\ stop$

$138 \leq 150$ ✓

بر خلاف نتایج آزمون‌ها زمانبندی RM، مشاهده می‌کنیم که بر اساس تحلیل RTA این مجموعه وظایف قابل زمانبندی است.

راه حل دوم:

چونکه در سوال گفته نشده که حتما باید از طریق تحلیل‌های زمانبندی RM به جواب برسید، می‌توانید به جای راه حل بالا نمودار این مجموعه وظایف را رسم کنید و قابلیت زمانبندی را اثبات کنید.

ب) ددلاین‌های آینده چطور؟ چگونه می‌توانید تضمین کنید که هیچ وقت ددلاینی رد می‌شد یا نه؟

راه حل اول:

الگوریتم RTA در واقع WCET هر کدام از این وظایف را محاسبه می‌کند به همین خاطر می‌توان اثبات کرد که اگر و تنها اگر مجموعه وظایفی بتواند شرایط این آزمون را برآورده کند، آنگاه این مجموعه وظایف حتما با الگوریتم RM قابل زمانبندی است.

راه حل دوم:

اگر در قسمت قبل نمودار رسم کرده‌اید در این قسمت باید Hyper period این مجموعه وظایف را به دست بیارید که و رسم نمودار را در حالت کلی برای این بازه انجام دهید.

Hyper period (100, 145, 150) = 8700

(نوشتن همین توضیحات برای این بخش کافی است و نیازی به رسم نمودار در این بازه نیست!)

سوال ۷:

با توجه به مجموعه وظایف داده شده به سوالات زیر پاسخ دهید. (نکته: در جواب به این سوال به فرضیات قضیه‌هایی که استفاده می‌کنید توجه کنید.)

	C_i	T_i	D_i
τ_1	2	5	5
τ_2	4	10	9
τ_3	1	25	25

۱) بررسی کنید که آیا این مجموعه وظایف شرط لازم را برای feasibility دارا است؟

$$U = (C1/T1) + (C2/T2) + (C3/T3)$$

$$U = (2/5) + (4/10) + (1/25)$$

$$U = 0.4 + 0.4 + 0.04 = 0.84 \leq 1$$

بله این مجموعه وظایف feasible است.

۲) با استفاده از آزمون زمانبندی بررسی کنید که آیا این مجموعه وظایف توسط الگوریتم EDF قابل زمانبندی است یا خیر.

در این مجموعه وظایف چونکه $D \neq T$ بنابراین نمی‌توان با استفاده از قاعده‌ی $U \leq 1$ تصمیم‌گیری کرد. باید از قاعده‌ی Processor Demand استفاده کرد که در بخش (۶) آن را انجام می‌دهیم.

۳) بررسی کنید که آیا این مجموعه وظایف آزمون L&L را با موفقیت پست سر می‌گذارد. از این آزمون چه نتیجه‌ای می‌گیرید؟

می‌توان این آزمون را برای این مجموعه وظایف نوشت ولی نمی‌توان از جواب آن هیچ نتیجه‌ای گرفت زیرا این مجموعه وظایف شرط اولیه آزمون L&L را که برابر بودن $T=D$ است را ندارد.

۴) Busy Window سطح سوم را محاسبه کنید.

در واقع با محاسبه‌ی WCRT وظیفه τ_3 می‌توان این مقدار را بدون رسم شکل بدست آورد.

$$R_3^{(0)} = 1$$

$$R_3^{(1)} = 1 + \left\lceil \frac{1}{5} \right\rceil \times 2 + \left\lceil \frac{1}{10} \right\rceil \times 4 = 1 + 2 + 4 = 7$$

$$R_3^{(2)} = 1 + \left\lceil \frac{7}{5} \right\rceil \times 2 + \left\lceil \frac{7}{10} \right\rceil \times 4 = 1 + 4 + 4 = 9$$
$$R_3^{(3)} = 1 + \left\lceil \frac{9}{5} \right\rceil \times 2 + \left\lceil \frac{9}{10} \right\rceil \times 4 = 1 + 4 + 4 = 9 \leq 9 \; stop$$

Busy-Window Level 3 = 9

۵) Critical Instance وظیفه‌ی τ_3 در چه زمانی رخ می‌دهد؟

Critical Instance وظیفه‌ی τ_3 زمانی رخ می‌دهد که وظیفه‌ی τ_3 هم زمان با دو وظیفه τ_1 و τ_2 وارد CPU شود. بنابراین برای اولین بار این اتفاق در t=0 رخ می‌دهد و بار بعدی در t=50 (ک.م.م سه تا دوره) است و به همین ترتیب در زمان‌های مضارب ۵۰ این اتفاق تکرار می‌شود.

۶) با استفاده از آزمون Processor Demand بررسی کنید که آیا روش EDF قادر است این مجموعه وظایف را زمانبندی کند.

Hyper period (5, 10, 25) = 50

$$L^* = \frac{(10 - 9) \times 0.4}{1 - 0.84} = \frac{0.4}{0.16} = 2.5$$

$$g_1(0,5) = \left(\left\lfloor \frac{5 - 5}{5} \right\rfloor + 1 \right) 2 = 2 \leq 5$$
$$g_2(0,9) = \left(\left\lfloor \frac{9 - 9}{10} \right\rfloor + 1 \right) 4 = 4 \leq 9$$
$$g_3(0,25) = \left(\left\lfloor \frac{25 - 25}{25} \right\rfloor + 1 \right) 1 = 1 \leq 25$$

بله، با توجه به اینکه کافی است این قضیه را برای L<2.5 چک کنیم و اولین ددلاین همه‌ی وظیفه‌ها بزرگ‌تر از ۲.۵ است (تا قبل از اولین ددلاین تمام مقادیر g صفر خواهد بود)، بنابراین بدون هیچ محاسبه‌ای می‌توان گفت که این مجموعه وظایف توسط الگوریتم EDF قابل زمانبندی است.

۷) آیا این مجموعه وظایف هارمونیک است؟ اگر جواب خیر است، با کمترین تغییرات آن را هارمونیک کرده و جدول وظایف جدید را رسم کنید.

خیر

	C_i	T_i	D_i
τ_1	2	5	5
τ_2	4	10	10
τ_3	1	30 یا 20	30 یا 20

سوال ۸:

می‌دانیم قضیه‌ی Processor Demand به صورت $\forall L > 0,\ g(0, L) \le L$ تعریف می‌شود. با استفاده از آزمون Processor Demand و با کمترین تعداد مرحله‌ی ممکن و کوچکترین L ممکن، بررسی کنید که آیا جدول مجموعه وظایف داده شده با استفاده از الگوریتم EDF قابل زمانبندی است یا خیر. فرض کنید که هر دو وظیفه در لحظه‌ی t=0 شروع به کار می‌کنند.

	C_i	T_i	D_i
τ_1	2	5	4
τ_2	3	7	5

Hyper period (5, 7) = 45

U = 0.4 + 0.428 ≈ 0.828

$$L^* = \frac{(5-4)\times 0.4 + (7-5)\times 0.428}{1-0.828} = \frac{1.256}{0.16} \approx 7.302$$

$$g_1(0,4) = \left(\left\lfloor \frac{4-4}{5} \right\rfloor + 1\right) 2 = 2 \le 4$$

$$g_2(0,5) = \left(\left\lfloor \frac{5-5}{7} \right\rfloor + 1\right) 3 = 3 \le 5$$

برای این قضیه کافیست L<7.302 را چک کنیم و به همین خاطر دیگر نیازی به محاسبه‌ی $g_1(0,9)$ و $g_2(0,12)$ نیست و می‌توان نتیجه گرفت که این مجموعه وظایف توسط الگوریتم EDF قابل زمانبندی است.

سوال ۹:

چهار task دوره‌ای، ۴ منبع (A, B, C, D) زیر را به صورت مشترک استفاده می‌کنند و بیشترین زمانی که هر task هر کدام از منابع را در اختیار دارد، در جدول زیر مشخص شده است. با توجه به جدول زیر و با استفاده از PIP، Maximum Blocking Time را برای هر کدام از taskها محاسبه کنید.

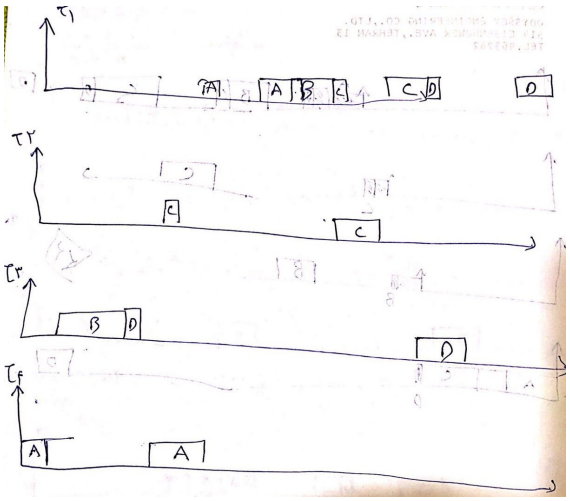
	A	B	C	D
τ_1	3	6	10	7
τ_2	0	0	8	0
τ_3	0	4	0	14
τ_4	7	0	9	11

فرض می‌کنیم که در این سوال اولویت تسک‌ها به این صورت است که تسک شماره ۱ بالاترین اولویت و تسک شماره ۴ کمترین اولویت را دارا است و سمافورها به ترتیب از A تا D در تسک ها اجرا می‌شوند.

با توجه به نوع الگوریتم ما که PIP است. برای هر تسک، سناریو متفاوتی قابل تعریف است.

می‌دانیم که هر وظیفه با اولویت پایین‌تر تنها یک بار می‌تواند وظیفه‌ای با اولویت بالاتر را بلاک کند. در نتیجه سناریو هر وظیفه مانند زیر است:

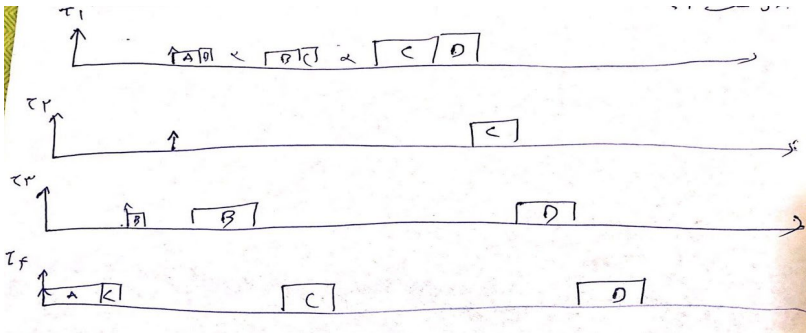
:Task1



با توجه به نمودارهای بالا، ابتدا وظیفه چهارم می‌خواهد از منبع A استفاده کند اما وظیفه سوم وارد می‌شود و از منبع B استفاده می‌کند. هنگام استفاده از منبع B وظیفه دوم وارد می‌شود و از منبع C استفاده می‌کند اما وظیفه اول وارد می‌شود. وظیفه اول زمانی که می‌خواهد از منبع A استفاده کند ابتدا باید وظیفه چهارم اجرا شود و بعد از آن می‌تواند از منبع A استفاده کند به همین ترتیب وظیفه‌های دوم و سوم نیز باید اجرا شوند.

در کل مجموع Blocking ها برابر است با ۷+۸+۱۴=۲۹ واحد زمانی.

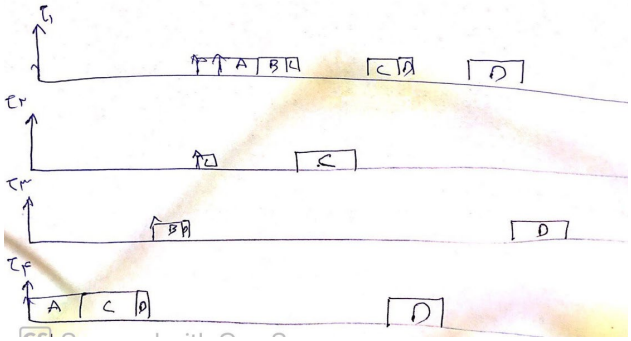
:Task2



ابتدا وظیفه چهارم وارد می‌شود و از منبع A استفاده می‌کند. زمانی که می‌خواهد از منبع C استفاده کند، وظیفه سوم وارد می‌شود و زمانی که می‌خواهد از منبع B استفاده کند وظیفه اول وارد می‌شود و از منبع A استفاده می‌کند. برای منبع B ابتدا باید وظیفه سوم اجرا شود و برای منبع C نیز باید ابتدا وظیفه چهارم اجرا شود.

در نتیجه ۴+۹=۱۳ مجموع زمان Blocking است.

:Task3



ابتدا وظیفه چهارم وارد شده و از منابع A ,C استفاده می‌کند. زمانی که می‌خواهد از منبع D استفاده کند، وظیفه سوم وارد شده و از منبع B استفاده می‌کند. در نهایت وظیفه دوم وارد می‌شود و زمانی که می‌خواهد از منبع C استفاده کند، وظیفه اول وارد می‌شود و از منبع A و B استفاده می‌کند؛ اما زمانی که می‌خواهد از منبع C استفاده کند، ابتدا باید وظیفه دوم اجرا شود. بنابراین وظیفه دوم از منبع C استفاده می‌کند و سپس وظیفه اول از آن استفاده می‌کند. زمانی که تسک اول می‌خواهد از منبع D استفاده کند، ابتدا باید وظیفه چهارم اجرا شود.

در نتیجه مجموع زمان Blocking برابر ۱۱ واحد است.

دقت کنید که استفاده تسک ۲ از منبع C برای تسک ۳ Blocking محاسبه نمی‌شود زیرا اولویت بالاتری از تسک ۳ دارد.

:Task4

مقدار Blocking ما صفر است زیرا هیچ وظیفه‌ای نمی‌تواند وظیفه چهارم را بلاک کند. زیرا تمامی آن‌ها اولویت بالاتری از وظیفه چهارم دارند.

در کل زمانی که برای وظیفه‌ای با بالاترین اولویت را محاسبه می‌کنیم، مقدار Blocking time ما از بقیه موارد بیشتر است.

سوال ۱۰:

دو مدل ارتباطی Conventional multi-wire looms و CAN bus network را در سیستم یک ماشین با یکدیگر مقایسه کنید و به سوالات زیر پاسخ دهید.

۱) دو مورد از مزیت‌های استفاده از CAN bus network را نسبت به Conventional multi-wire looms بیان کنید.

الف) Reduced Weight

ب) Reduced Cost

ج) Increased Reliability

۲) دو مورد از مزیت‌های استفاده از Conventional multi-wire looms نسبت به CAN bus network را بیان کنید.

الف) سریعتر است. چون در CAN bus در هر لحظه فقط یک دو بخش می‌توانند بایکدیگر ارتباط داشته باشند و بقیه بخش‌ها باید صبر کنند ولی در multi-wire همه می‌توانند در لحظه با هم صحبت کنند و تداخلی وجود ندارد.

ب) طراحی پروتکل ساده‌تری دارد. دقیقا به دلیل گفته شده در بالا، چونکه در CAN bus همه نمی‌توانند هم زمان صحبت کنند بنابراین باید یک اولویت‌بندی برای آن تعیین کرد.

ج) اگر قسمتی از سیم کشی در CAN bus قطع شود، کل سیستم مختل می‌شود. اما در multi-wire اگر قسمتی از سیم‌کشی قطع شود، فقط ارتباط دو بخشی که در دو سر سیم است مختل می‌شود و بقیه بخش‌ها مشکلی ندارند و به می‌توانند به صحبت خود ادامه دهند.

سوال ۱۱:

فرض کنید در پروتکل CAN گره (۱) در حال ارسال رشته بیت 001011101 است و در کلاک‌های $C_1C_2C_3$ بیت‌های 001 ارسال شده‌اند. اگر در کلاک چهارم گره‌های (۲) و (۳) به ترتیب آماده‌ی ارسال رشته بیت‌های 000001010 و 110110110 باشند، در کلاک‌های $C_4C_5C_6$ چه اطلاعاتی در گذرگاه مشترک (Bus) ارسال خواهد شد؟

	C_1	C_2	C_3	C_4	C_5	C_6
1	0	0	1	0	1	-
2	-	-	-	0	0	0
3	-	-	-	1	-	-

$C_4C_5C_6 = 000$